Filtering (Exercices)

1 Filtering Constraints

1.1 Filtering `allDifferent`

1. For a Sudoku block, we have the following constraint `allDifferent`$(x_{00}, x_{01}, \ldots, x_{22})$:

   Sudoku Block

3	12	29
156	8	26
679	4	19

   Which values can be removed when enforcing AC?

2. For a Sudoku block, we have the following constraint `allDifferent`$(x_{00}, x_{01}, \ldots, x_{22})$:

   Sudoku Block

3	126	1256
156	8	126
1256	4	125 679

   Which values can be removed when enforcing AC? What can you conclude?

1.2 Filtering `sum`

1. Let the constraint:

   $$2x_1 + 3x_2 + 2x_3 + x_4 + 4x_5 + 2x_6 \geq 50$$

   with $\forall i \in 1..6, dom(x_i) = \{1, 2, 3, 4\}$.

   Which values can be removed when enforcing AC?

2. Let the constraint:

$$12 \geq 2x_1 + 2x_2 + x_3 \geq 9$$

with $\forall i \in 1..3, dom(x_i) = \{1, 2\}$.

Which values can be removed when enforcing AC? You will build a Knapsack graph, its reduced version and the corresponding MDD.

1.3 Filtering Arithmetic Constraints

Let the ternary constraint $x < y + z$

1. Identify BC filtering rules for each variable with respect to the min and max bounds of the other variables (in order to establish AC).
2. Assuming that $dom(x) = \{4, 5, 6, 7, 8\}$, $dom(y) = \{0, 1, 2, 3\}$ and $dom(z) = \{2, 3\}$, indicate which values are removed when enforcing AC on $x < y + z$.

1.4 Filtering Logical Combinations

1. Let x and y be two variables such that $dom(x) = dom(y) = \{1, 2, \ldots, 10\}$. Indicate what is the AC filtering achieved by the meta-constraint:

$$\texttt{or}(x + 7 \leq y, y + 6 \leq x)$$

2. Let w, x, y and z four variables such that $dom(w) = dom(x) = dom(y) = dom(z) = \{a, b, c\}$. Indicate what is the AC filtering achieved by the meta-constraint:

$$\texttt{and}(c_{wxy}, c_{xyz})$$

where (the relations of) c_{wxy} and c_{xyz} are defined as follows:

w	x	y
a	b	c
b	c	a
c	b	b
c	c	b
a	a	a

(a) $rel(c_{wxy})$

x	y	z
b	a	a
c	a	b
b	c	a
a	b	c
a	a	c

(b) $rel(c_{xyz})$

1.5 Filtering Bounds

Assuming that domains are totally ordered by a relation $<$, the minimum and maximum values of the current domain of a variable x are respectively denoted by $min(x)$ and $max(x)$. For example, for $dom(x) = \{2, 3, 5, 8\}$, $min(x)$ and $max(x)$ are equal to 2 and 8, respectively.

A *bound support* on a constraint c is a tuple τ such that $\tau \in rel(c)$ and $\forall x \in scp(c)$, $min(x) \leq \tau[x] \leq max(x)$. A *support* on a constraint c is a tuple τ such that $\tau \in rel(c)$ and $\forall x \in scp(c)$, $\tau[x] \in dom(x)$.

- A constraint c is bound(Z)-consistent, or BC(Z), iff $\forall x \in scp(c)$, $(x, min(x))$ and $(x, max(x))$ belong to a bound support on c.

- A constraint c is bound(D)-consistent, or BC(D), iff $\forall x \in scp(c)$, $(x, min(x))$ and $(x, max(x))$ belong to a support on c.
- A constraint c is range-consistent, or RC, iff $\forall x \in scp(c)$, $\forall a \in dom(x)$, (x, a) belongs to a bound support on c.

Let P be a CN such that:

- $vars(P) = \{x_1, \ldots, x_6\}$ where:
  - $dom(x_1) = \{1, 2\}$,
  - $dom(x_2) = \{1, 2\}$,
  - $dom(x_3) = \{2, 3, 5, 6\}$,
  - $dom(x_4) = \{2, 3, 5, 6\}$,
  - $dom(x_5) = \{5\}$,
  - $dom(x_6) = \{3, 4, 5, 6, 7\}$,
- $ctrs(P) = \{\texttt{allDifferent}(x_1, x_2, x_3, x_4, x_5, x_6)\}$

Give the result (state of domains) of applying on P an algorithm enforcing:

1. BC(Z)
2. BC(D)
3. RC
4. AC

1.6 Filtering Multi-valued Decision Diagrams

You have to write an algorithm that enforces (generalized) arc consistency on MDD constraints. We will use an approach similar to STR, recording arc-consistent values while exploring the structure of the constraint (here, an MDD). The skeleton of the algorihtm is:

Algorithm 0: enforceAC(c: MDD Constraint)

```
foreach variable x ∈ scp(c) do
    gacValues[x] ← ∅
exploreMDD(mdd(c))          // gacValues updated during exploration
// Domains are now updated
foreach variable x ∈ scp(c) do
    dom(x) ← gacValues[x]
```

Concerning notations, we shall use:

- $mdd(c)$ for denoting the root node of the MDD associated with a constraint c
- $sink(c)$ for denoting the sink node of the MDD associated with a constraint c
- with ν denoting a node in the MDD,
  - $var(\nu)$ denotes the variable associated with the level of the node ν

- $arcs(\nu)$ denotes the set of outgoing arcs from node ν

- with α denoting an arc in the MDD,
  - $label(\alpha)$ denotes the value labelling the arc α
  - $target(\alpha)$ denotes the node that represents the destination of the arc α

1. in a first step, write a basic algorithm
2. in a second step, avoid exploring several times the same nodes during exploration

2 Constraint Propagation

2.1 Exercice

The following code describes a model written in PyCSP^3. Give the AC-closure of this constraint network.

```
from pycsp3 import *

x = Var(range(6))
y = Var(range(6))
z = Var(range(6))

satisfy(
    x > y,
    x != z,
    y > z
)
```

2.2 Exercice

The following code describes a model written in PyCSP^3. Simulate the process of constraint propagation (AC-closure) on this constraint network.

```
from pycsp3 import *

x = Var(1,2,3,4)
y = Var(2,3,4)
z = Var(2,3)

satisfy(
    x + 2*y - z <= 4,
    AllDifferent(x,y,z)
)
```