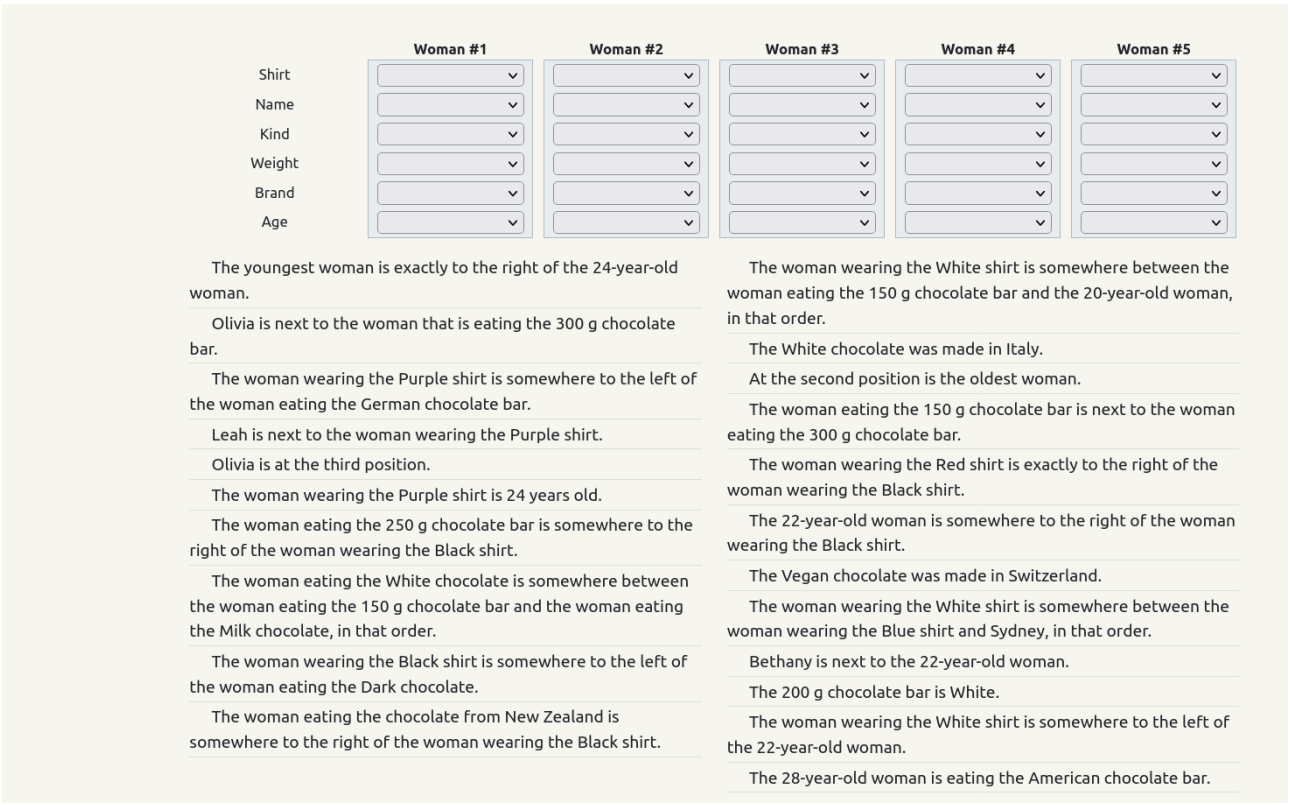
UE ASTRE - Mini Projet SMT

Victor Estevez

1. Zebra puzzle

Chocolate Bars Zebra Puzzle

Five women are side by side eating their favorite chocolate bars. The chocolates have different kinds, weights and brands (designated by their countries). Find out who is eating the organic chocolate bar.



	Woman #1	Woman #2	Woman #3	Woman #4	Woman #5
Shirt					
Name					
Kind					
Weight					
Brand					
Age					

- The youngest woman is exactly to the right of the 24-year-old woman.
- Olivia is next to the woman that is eating the 300 g chocolate bar.
- The woman wearing the Purple shirt is somewhere to the left of the woman eating the German chocolate bar.
- Leah is next to the woman wearing the Purple shirt.
- Olivia is at the third position.
- The woman wearing the Purple shirt is 24 years old.
- The woman eating the 250 g chocolate bar is somewhere to the right of the woman wearing the Black shirt.
- The woman eating the White chocolate is somewhere between the woman eating the 150 g chocolate bar and the woman eating the Milk chocolate, in that order.
- The woman wearing the Black shirt is somewhere to the left of the woman eating the Dark chocolate.
- The woman eating the chocolate from New Zealand is somewhere to the right of the woman wearing the Black shirt.
- The woman wearing the White shirt is somewhere between the woman eating the 150 g chocolate bar and the 20-year-old woman, in that order.
- The White chocolate was made in Italy.
- At the second position is the oldest woman.
- The woman eating the 150 g chocolate bar is next to the woman eating the 300 g chocolate bar.
- The woman wearing the Red shirt is exactly to the right of the woman wearing the Black shirt.
- The 22-year-old woman is somewhere to the right of the woman wearing the Black shirt.
- The Vegan chocolate was made in Switzerland.
- The woman wearing the White shirt is somewhere between the woman wearing the Blue shirt and Sydney, in that order.
- Bethany is next to the 22-year-old woman.
- The 200 g chocolate bar is White.
- The woman wearing the White shirt is somewhere to the left of the 22-year-old woman.
- The 28-year-old woman is eating the American chocolate bar.

- script SMT :

(les scripts complets pouvant être assez long, je ne garde que des extraits dans ce pdf)

```
(define-fun domaine ((x Int)) Bool  (and (> x 0) (< x 6)))
;;Shirt:
(declare-const S1 Int) ;Black
(declare-const S2 Int) ;Blue
[...]
;;Kind:
(declare-const K1 Int) ;dark
(declare-const K2 Int) ;milk
[…]
;;On s'assure que chaque catégorie n'est attribué qu'à une femme
(assert(distinct S1 S2 S3 S4 S5))
;;Et qu'on ai bien 5 que personnes
(assert (domaine S1))
(assert (domaine S2))
```

[...]
;;Traduction des contraintes
;;The youngest woman(A1) is exactly to the right of the 24-year-old woman(A3).
(assert (= A1 (+ 1 A3)))
;;Olivia(N4) is next to the woman that is eating the 300 g chocolate bar(W5).
(assert (or (= -1 (- N4 W5)) (= 1 (- N4 W5))))
[…]
(check-sat)
(get-model)

J’avais au départ mit toutes les variable à un type énuméré Woman mais j’ai du les changer en Int pour pouvoir utiliser les fonctions (>) et (<) pour les contraintes de type à gauche/à droite.
Du coup ajout d'une fonction domaine + un assert domaine pour toutes les variables.
On pourrait rajouter des fonctions pour chaque type de contraintes qui demandes plus qu’une fonction de base, pour factoriser le code, par exemple :
Olivia(N4) is next to the woman that is eating the 300 g chocolate bar(W5)
Donnerait : (define-fun isnextto ((x Int) (y Int)) Bool (or (= -1 (- x y)) (= 1 (- x y))))
qu’on utiliserait avec (assert (isnextto N4 W5))

Pour le Z3 output:
Z3 output

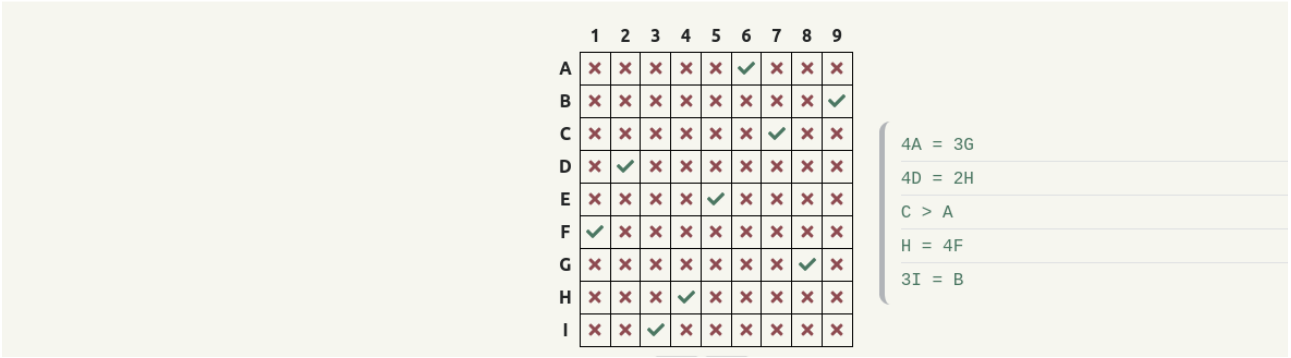
```
     sat
     (model
       (define-fun K2 () Int
         4)
       (define-fun S5 () Int
         2)
[...]
       (define-fun A1 () Int
         4)
     )
```

(define-fun K2 () Int 4) signifie que ourla dame #4 doit manger le chocolat au lait (K2).
Les valeurs du Z3 output résolves effectivement le problème une fois entrée sur la page du puzzle, et pour répondre à la question du sujet, le chocolat organique est mangé par le dame #2.
Après vérification avec des (assert(not(= K2 4))), on a une solution unique.

2. Logic Equation

Logic Equations 9x9 - #10

In this 9x9 Logic Equations you have to find unique integer values for the variables (ranging from 1 to 9) to make all statements true.



-Script SMT :

(define-fun domaine ((x Int)) Bool (and (> x 0) (< x 10)))
(declare-const A Int)
(declare-const B Int)
[...]
(declare-const I Int)
;;you have to find unique integer values for the variables :
(assert(distinct A B C D E F G H I))
;;ranging from 1 to 9 :
(assert (domaine A))
[...]
(assert (domaine I))

(assert (= (* 4 A) (* 3 G)))
(assert (= (* 4 D) (* 2 H)))
(assert (> C A))
(assert (= H (* 4 F)))
(assert (= B (* 3 I)))

(check-sat)
(get-model)

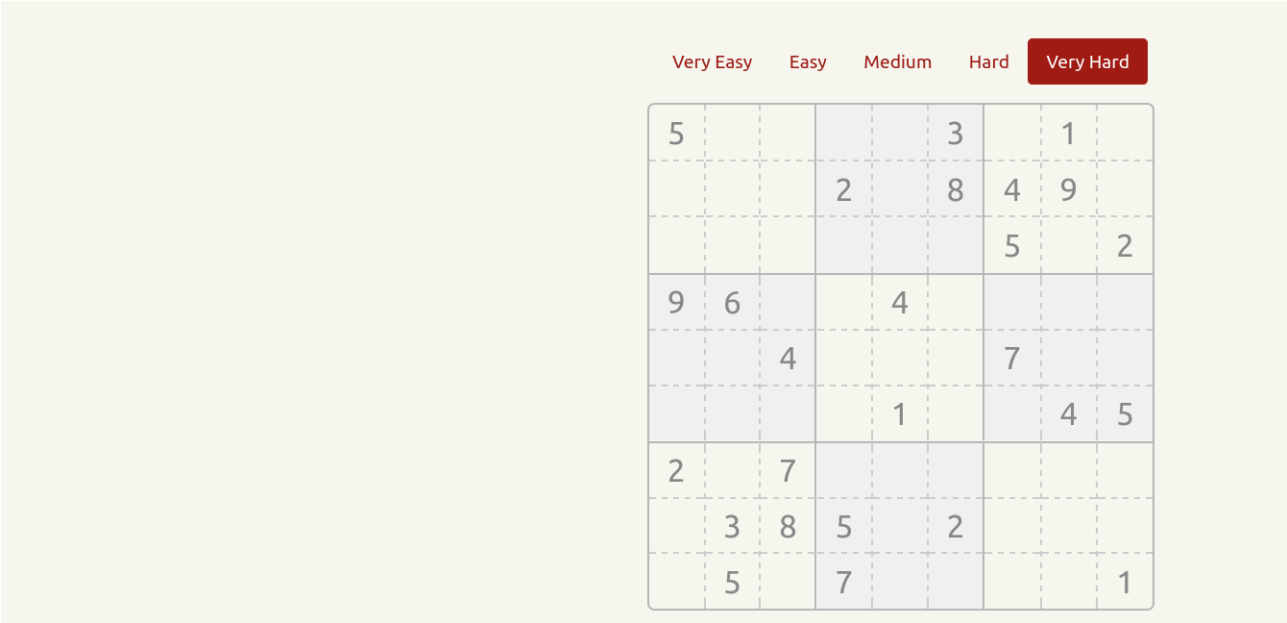
```
Z3 output :
sat
(model
  (define-fun A () Int    6)
[...]
  (define-fun H () Int    4)
)
```

Pas grand-chose à ajouter, ce puzzle s'adapte parfaitement au format SMT, on entre simplement toutes les équations 1:1.
Encore une fois la solution et unique.

3. Sudoku

Sudoku

Sudoku is very entertaining logic puzzle. It became popular in the early 00's and since then is still a good game for all ages and skills. The objective is to fill a 9×9 grid with digits from 1 to 9 following some simple rules.



-Script SMT :

(declare-datatypes () ((Val V1 V2 V3 V4 V5 V6 V7 V8 V9)))

;;Correspond à une case du sudoku ex: C2 est la case 3eme ligne 2eme colonne :
(declare-const A1 Val)
(declare-const A2 Val)
[...]
(declare-const I8 Val)
(declare-const I9 Val)

;;Les lignes ont des valeurs distinctes:
(assert(distinct A1 A2 A3 A4 A5 A6 A7 A8 A9))
[...]
(assert(distinct I1 I2 I3 I4 I5 I6 I7 I8 I9))

;;Les colonnes ont des valeurs distinctes:
(assert(distinct A1 B1 C1 D1 E1 F1 G1 H1 I1))
[...]
(assert(distinct A9 B9 C9 D9 E9 F9 G9 H9 I9))

;;Les carrés 3x3 ont des valeurs distinctes:
(assert(distinct A1 A2 A3 B1 B2 B3 C1 C2 C3))
(assert(distinct D1 D2 D3 E1 E2 E3 F1 F2 F3))
[…]
(assert(distinct D7 D8 D9 E7 E8 E9 F7 F8 F9))
(assert(distinct G7 G8 G9 H7 H8 H9 I7 I8 I9))

;;On fige les numéros déjà présent dans la grille:
(assert (= A1 V5))
[...]
(assert (= I9 V1))

(check-sat)
(get-model)

Z3 output

```
sat
(model
  (define-fun C5 () Val
    V7)
[...]
  (define-fun A3 () Val
    V2)
)
```

Le model nous donne la valeur de chaque case, elles completent effectivement le sudoku.
Comme on ne fait pas de comparaison entre les variables dans ce puzzle, on peut donc utiliser un type énuméré et s’épargner les assert domaine sur les 81 cases.