**Metodos Formais em Engenharia de Software (2023/2024)**

# SMT solvers

**Exercício 1 (Criptograma)** O Cryptarithms é um jogo que consiste numa equação matemática entre números desconhecidos, cujos dígitos são representados por letras. Cada letra deve representar um dígito diferente e o dígito inicial de um número com vários dígitos não deve ser zero.

Use o Z3 para o ajudar a descobrir os dígitos a que correspondem as letras envolvidas na seguinte equação: **SEND + MORE = MONEY**

*Sugestão:* Escreva diretamente a equação, representado cada parcela por uma expressão aritmérica onde cada letra é multiplicada pelo seu "peso específico" em base 10.

Confirme que só existe uma solução para este puzzle.

**Exercício 2 (Codificação lógica de um programa)** Considere o seguinte programa C sobre inteiros.

```
z = 0;
x = x + y;
if (y >= 0) {
   y = x - y;
   x = x - y;
}
else {
   z = x - y;
   x = y;
   y = 0;
}
z = x + y + z;
```

1. Faça a codificação lógica deste programa, recordanao que nesse processo, o programa deverá ser transformado em formato *single-assignment (SA)* e depois em *conditional normal form (CNF)*.

2. Tendo por base a codificação lógica que fez do programa, utilize o SMT solver Z3 para se pronunciar quanto à veracidade das afirmações abaixo indicadas. No caso da afirmação ser falsa, apresente o contra-exemplo indicado pelo solver.

   (a) *"'Se o valor inicial de y for positivo, o programa faz a troca dos valores de x e y entre si."*

   (b) *"'O valor final de y nunca é negativo."*

   (c) *"O valor final de z corresponde à soma dos valores de entrada de x e y."*

   (d) *"O valor final de x é sempre negativo."*

**Exercício 3 (Codificação lógica de um programa)** Recorde o seguinte programa C, já apresentado na aula teórica.

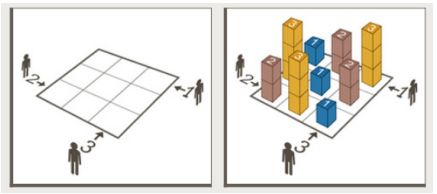
```
if (x > 0)
{
  x = x - 10;
  if (y < 0)
     x = x - y;
  else
     x = x + y;
}
r = x + x;
```

Faça a codificação lógica do programa no Z3 e use o solver a para se pronunciar quanto à veracidade das afirmações que se seguem. No caso da afirmação ser falsa, apresente o contra-exemplo indicado pelo solver.

1. *"O valor final de x é sempre positivo."*
2. *"Se o valor inicial de x for superior a 15, no final do programa, r é superior x."*

**Exercício 4 (Skyscrapers)**

O Skyscrapers é um puzzle lógico que tem por objectivo organizar arranha-céus num tabuleiro ($N \times N$) de forma a que o seu horizonte seja visível de acordo com as pistas (números colocados nas bordas do tabuleiro que indicam quantos arranha-céus é possível ver daquela posição.



Adicionalmente, exige-se que:

- Todos os arranha-céus têm altura entre 1 e $N$.
- Não podem existir arranha-céus da igual altura numa mesma coluna ou linha.
- O tabuleiro está inicialmente vazio.

Tendo em conta as explicações dadas acima, exprima as restrições necessárias e resolva o problema para tabuleiros de dimensão 3×3, usando o Z3 com a teoria de inteiros.

*Sugestão:* Lembre-se que pode usar expressões condicionais. Na API do Z3 estas expressões têm a seguinte sintaxe `If(.,.,.)` Por exemplo, usando expressões condicionais, podemos criar uma função Python que recebe três argumentos (uma fila de arranha-céus) $a_1$, $a_2$ e $a_3$ e devolve o número de prédios visíveis quando olhamos assim: $\rightarrow a_1\ a_2\ a_3$.

```
def visible(a1,a2,a3):
    return If(a1==3, 1, If(a2==3, 2, If(a1==1, 3, 2)))
```

Depois de codificar as regras do puzzle, acrescente as restrições correspondentes à definição de um tabuleiro concreto (por exemplo, o da figura) e apresente a solução gerada pelo solver.