

# Especificação e Modelação

## Exercícios sobre SMV

1. Numa colónia de camaleões cada um dos camaleões pode, em cada momento, ter uma das seguintes cores: vermelho, azul ou verde. Sempre que dois camaleões de cores diferentes se encontram, ambos mudam para a cor que nenhum deles tinha. Caso contrário não mudam de cor. O objectivo deste puzzle é descobrir uma sequência de encontros entre os camaleões que faça a colónia convergir toda para a mesma cor. O seguinte modelo (incompleto) SMV especifica este puzzle, para uma colónia com 6 camaleões, em que 1 deles é vermelho (cor 0), 2 são azuis (cor 1) e 3 são verdes (cor 2). As *ivars* *i* e *j* representam os identificadores dos camaleões que se encontram em cada momento.

```
MODULE
  main
VAR
  cor : array 0..5 of 0..2;
IVAR
  i : 0..5;
  j : 0..5;
ASSIGN
  init(cor[0]) := 0;
  init(cor[1]) := 1;
  init(cor[2]) := 1;
  init(cor[3]) := 2;
  init(cor[4]) := 2;
  init(cor[5]) := 2;
```

- (a) Complete este modelo, especificando como evolui o array que representa as cores dos camaleões. Note que, quando os dois camaleões que se encontram tem cores diferentes, a nova cor pode ser calculada subtraindo a soma das duas cores a 3.
- (b) Mostre como poderia usar o SMV para descobrir se existe uma sequência de encontros que faça a colónia convergir para uma única cor.
- (c) Explique o que representa o seguinte `aux` no contexto deste modelo.

```
DEFINE
  aux := (cor[0]=0 ? 1 : 0) + (cor[1]=0 ? 1 : 0) + (cor[2]=0 ? 1 : 0) +
         (cor[3]=0 ? 1 : 0) + (cor[4]=0 ? 1 : 0) + (cor[5]=0 ? 1 : 0);
```

- (d) Indique poderia modificar o modelo anterior e verificar usando o SMV que, sempre que no estado inicial o número de camaleões de duas das cores diferem entre si num múltiplo de 3 unidades, então é possível fazer convergir a colónia para uma única cor.

2. Considere a sequência de inteiros definida pela seguinte função de iteração:

```
next(n) := n mod 2 = 0 ? n/2 : n*3+1;
```

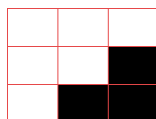
A conjectura de Collatz diz que esta sequência, começando num número *n* qualquer estritamente positivo, inevitavelmente vai chegar ao valor 1. Mostre como pode usar o SMV para

verificar uma versão “bounded” desta conjectura para números  $n$  entre 1 e 1000. Note que neste caso a expressão  $n*3+1$  pode fazer *overflow*, pelo que a conjectura de Collatz tem que ser adaptada para: se não existir overflow, então a sequência vai inevitavelmente chegar a 1.

3. O jogo **Lights Out** consiste numa grelha  $5 \times 5$  de lâmpadas. Quando o jogo começa um sub-conjunto aleatório das lâmpadas está acesa. Ao pressionar uma das lâmpadas o seu estado, assim como das adjacentes, muda (uma lâmpada acesa fica apagada e uma apagada fica acesa). O objectivo do jogo é desligar todas as lâmpadas. Mais informação em

<http://www.logicgamesonline.com/lightsout/>

- (a) Mostre como poderia usar o SMV para resolver o seguinte puzzle (quadrados brancos são luzes ligadas)? Indique também qual a solução obtida.



- (b) Como poderia modificar o modelo anterior para verificar também que o número mínimo de acções para resolver o puzzle é 4?
- (c) É possível verificar usando o SMV que todos os puzzles  $3 \times 3$  tem solução? Em caso afirmativo, explique como.
4. Mostre como poderia usar o SMV para resolver o puzzle da ponte e da tocha, descrito na seguinte página da Wikipedia:

[http://en.wikipedia.org/wiki/Bridge\\_and\\_torch\\_problem](http://en.wikipedia.org/wiki/Bridge_and_torch_problem)

5. Mostre como poderia usar o SMV para resolver o conhecido puzzle das rãs ilustrado pela seguinte imagem.



Num charco temos uma sequência de 7 pedras alinhadas, 3 rãs verdes encostadas ao lado esquerdo e 3 rãs castanhas encostadas ao lado direito. Em cada pedra pode estar no máximo uma rã. Cada rã pode mover-se para a próxima pedra ou saltar por cima de uma rã de cor diferente que esteja imediatamente à sua frente. O objectivo do puzzle é trocar a posição das rãs verdes com as castanhas. Para perceber melhor o puzzle pode tentar resolver o mesmo no seguinte *site*:

<http://www.digyourowngrave.com/frog-jumping-puzzle/>

6. Considere o problema clássico de implementar uma máquina de trocar uma certa quantia em moedas. Modele esta máquina em SMV e indique como pode usar o respectivo model checker para determinar a sequência de moedas que troca uma certa quantia. Assuma que existe um stock limitado de moedas com as seguintes denominações: 1, 3 e 4.

7. Modele em SMV o algoritmo de exclusão mútua de Dekker e verifique se as propriedades de exclusão mútua e *no starvation* se verificam nesse algoritmo. A descrição do algoritmo pode ser encontrada na respectiva página da Wikipedia:

[http://en.wikipedia.org/wiki/Dekker%27s\\_algorithm](http://en.wikipedia.org/wiki/Dekker%27s_algorithm)