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 5×5 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 acessa). 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.

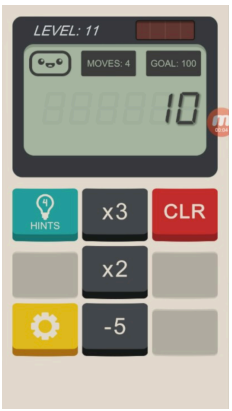
   (b) Como poderia modificar o modelo anterior para verificar também que o número mínimo de acções para resolver o puzzle é 4?

   (c) É possível verificar usando o SMV que todos os puzzles 3×3 tem solução? Em caso afirmativo, explique como.

4. Uma máquina de bebidas serve café e chá. O café custa 3 moedas e o chá 2 moedas. A máquina aceita no máximo 10 moedas, podendo o utilizador seleccionar café ou chá quando tiver inserido moedas suficientes. Nesse momento a porta da máquina abre-se podendo o utilizador retirar a bebida seleccionada. Depois de retirada a bebida a porta volta a fechar-se, podendo só então o utilizador voltar a seleccionar outra bebida. Em qualquer momento o utilizador pode inserir mais moedas.

   - Modele esta máquina em NuSMV.
   - Especifique fórmulas em lógica temporal que permitam verificar (ou refutar) as seguintes propriedades:
     - É sempre possível gastar todo o saldo.
     - Nunca é possível a porta ficar indefinidamente aberta.
     - Se o saldo é 0 e a porta está fechada, só se volta a abrir depois de terem sido inseridas pelo menos duas moedas.

5. O jogo *Calculator* é um puzzle que tem por objectivo encontrar a sequência de operações aritméticas que dará origem a um valor objectivo. Por exemplo, o nível 11 deste jogo é o seguinte.

Neste nível o valor inicial é 10, o objectivo é 100, e apenas pode executar 4 operações a escolher de entre as 3 disponíveis (multiplicar por 3, multiplicar por 2 ou subtrair por 5). Uma possível solução é começar por multiplicar por 3, depois subtrair 5, e finalmente multiplicar 2 vezes por 2.

(a) Mostre como poderia usar o NuSMV para encontrar uma solução para este nível do *Calculator*. Assuma que a calculadora apenas permite representar números entre 0 e 999.

(b) Especifique fórmulas em lógica temporal que permitam verificar (ou refutar) as seguintes propriedades:

- É possível bloquear a calculadora no valor 0.
- O valor da máquina só pode ser superior a 10 depois de escolhida uma operação de multiplicar.
- A solução acima apresentada é a única que permite resolver o nível 11.

6. Considere o seguinte programa em C que fica indefinidamente a tentar ordenar um array usando trocas sucessivas.

```
int a[5];
int i;
while (1) {
  for (i=0;i<4;i++) {
    if (a[i] > a[i+1]) swap(a,i,i+1);
  }
}
```

(a) Mostre como poderia usar o NuSMV para modelar este programa. Considere que o array só pode conter valores entre 0 e 9 e que a operação de *swap* é atómica.

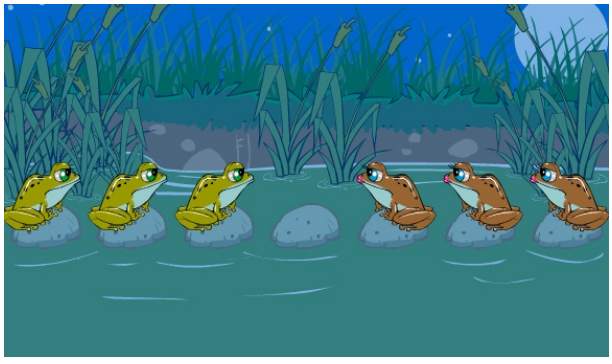
(b) Especifique fórmulas em lógica temporal que permitam verificar (ou refutar) as seguintes propriedades:

- O array vai inevitavelmente ficar ordenado.
- O array está desordenado até um momento em que fica ordenado para sempre.
- O ciclo `while` vai executar indefinidamente.

7. 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`

8. 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/`

9. 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.

10. 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`