Análise, Modelação e Teste

18 de Fevereiro de 2010

1. (35%) Considere o seguinte modelo de um gestor de memória. A relação `free` contém o conjunto de endereços livres. A relação `blocks` representa os blocos alocados: um bloco é essencialmente uma sequência de endereços, apontado pelo primeiro deles.

```
open util/ordering[Addr]
open util/integer

sig Addr {}

sig Memory {
   free : set Addr,
   blocks : Addr -> Addr
}

pred Inv [m : Memory] { ... }

pred Free [m, m' : Memory, a : Addr] { ... }

pred Alloc [m, m' : Memory, n : Int, a' : Addr] { ... }
```

   (a) Como poderia especificar no predicado `Inv` os seguintes invariantes?
      i. Não existem endereços simultanemaente livres e ocupados.
      ii. Um endereço não pode estar alocado em dois blocos diferentes.
      iii. O apontador para um bloco faz parte dos endereços alocados nesse bloco.
      iv. Os endereços de um bloco são sequenciais e começam no apontador para o mesmo.

   (b) Especifique a operação `Free` que liberta os endereços do bloco apontado por `a`.

   (c) Como pode verificar a correcção e consistência desta operação?

   (d) Especifique a operação `Alloc` que aloca `n` endereços e retorna o apontador `a'` para o novo bloco.

2. (10%) Considere que existe um sistema de transição associado a este gestor de memória, onde no estado inicial todos os endereços estão livres e as transições de estado resultam das operações `Free` e `Alloc`.

```
fact {
   first.free = Addr
   no first.blocks
   all m : Memory, m' : m.next {
     some a : Addr | free[m,m',a] or
     some n : Int, a' : Addr | alloc[m,m',n,a']
   }
}
```

Supondo que o Alloy suporta especificação de propriedades temporais usando lógica CTL, especifique as seguintes propriedades sobre os traços deste sistema de transição. Assuma que o estado está sempre acessível na variável `m`: por exemplo, a invariância poderia ser especificada como `AG Inv[m]`.

- Nunca são alocados mais do que 3 endereços.
- Pelo menos um endereço vai necessariamente ser alocado.
- Não é possível haver dois estados consecutivos sem blocos alocados.
- É sempre possível libertar todos os endereços alocados.

3. (15%) Considere o seguinte fragmento de uma classe Java para implementar o gestor de memória modelado em Alloy:

```
public class Memory {
  private static int MAX=20;
  public int[MAX] addr;
  public void free(int a) { ... } throws NullPointer;
}
```

Aqui é usado um array de inteiros para gerir tanto os endereços livres como os blocos alocados, sendo `MAX` o tamanho da memória. Sempre que um endereço está livre o conteúdo do array nessa posição é um número negativo. Senão contém o endereço do bloco ao qual pertence. Usando JML:

   (a) Anote esta classe com os invariantes que julgue adequados, tendo em atenção os invariantes previamente especificados em Alloy e outros inerentes a esta implementação concreta.

   (b) Especifique o método `free` que liberta os endereços do bloco apontado por `a`. Considere para o efeito que é lançada a excepção `NullPointer` se o endereço é bem formado mas não aponta para um bloco.

4. (15%) MPlayer (`http://www.mplayerhq.hu/`) is an open source media player available on all major operating systems. The main programming language used for MPlayer is `C`. Source code metrics have been derived for MPlayer at the file level and at the method level. They can be found in the spreadsheet `MPlayerMetrics.xls`.

   (a) In order to assess the quality of the source code of MPlayer, create a quality profile for complexity at the method level. Use the following guidelines:
   - Complexity risk categories are: *low risk* (1-10), *moderate risk* (11-20), *high risk* (21-50), and *very high risk* (50+).
   - Sum the volume (lines of code) per risk category.
   - Derive the relative volume (percentage) per risk category.
   - Create a pie-chart of the percentages.

   (b) What should the risk categories be for lines of code (LOC) at the file level, if we want at most 10% of the volume (lines of code) of the system to fall into the very high risk category, 10% of the volume in the high risk category, and 10% of the volume in the moderate risk category? Take the following steps:
   - Sort descending by LOC.
   - Compute the relative volume (percentage) for each file.
   - Complete the following table: *low risk* (0 LOC - ... LOC), *moderate risk* (... LOC - ... LOC), *high risk* (... LOC - 2240 LOC), *very high risk* (2240 LOC or larger).

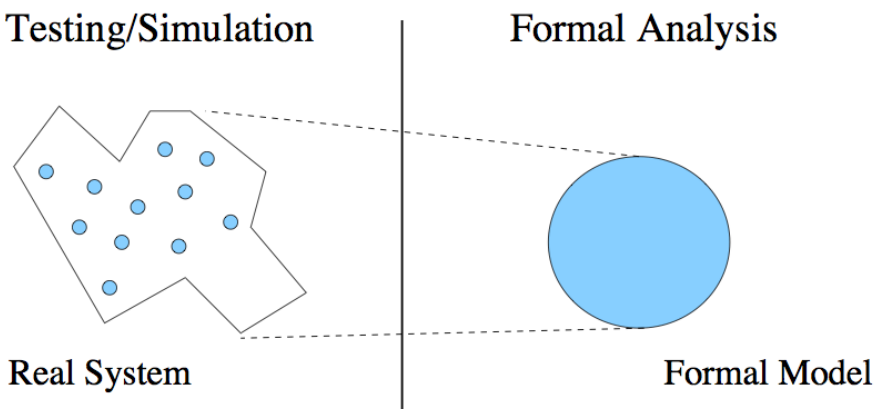
5. (15%) Considere o seguinte fragmento de um programa:

```
1:      Bool A,B,C; Int V;
2:      read(A);read(B);read(C);read(V);
3:      while (V<20)  {
4:              if ( A /\ ( B \/ C ) )
5:                      V:= V+5;
6:              if (A /\ ~B)
7:                      V:=V+10;
8:      }
9:      print V;
```

   (a) Defina um conjunto mínimo de casos de teste para cobrir todas as instruções, condições e decisões. Justifique a cobertura.

   (b) Os casos de teste da alínea anterior garantem cobertura modificada de condições e decisões (MC/DC - *multiple condition decision coverage*)? Justifique.

6. (10%) Numa recente palestra sobre métodos formais, John Rushby apresentou a seguinte figura para ilustrar a relação entre os métodos formais e o teste:



   Comente esta figura à luz das técnicas e ferramentas que estudou neste módulo.