CORREÇÃO DE PROGRAMAS - Aula 4 (http://frama-c.com/)

A teoria sobre correção de algoritmos já está, hoje em dia, implementada em ferramentas que permitem a verificação de código real, devidamente acrescido de anotações. Como exemplo do uso prático dos conceitos e técnicas apresentados nas notas de aula anteriores, nós utilizaremos a plataforma Frama-C para a verificação de pequenos programas escritos em C:

  http://frama-c.com/

As referências ao final listam materiais para informações adicionais.

UM PRIMEIRO EXEMPLO

Considere o seguinte algoritmo:

```
  ========================================
  Algoritmo: valor_absoluto
  Entrada: um número inteiro "x".
  Saída: um número inteiro.
  ----------------------------------------
  1. SE x < 0
  2. | RETORNE -x.
  3. RETORNE x.
  ========================================
```

Em C, uma função equivalente é:

```
  ========================================
  int valor_absoluto (int x)
    {
    if (x < 0)
      { return -x; }
    return x;
    }
  ========================================
```

Para verificá-la, precisamos conhecer a especificação da correção da função, que podemos estipular como sendo a seguinte:

```
  ----------------------------------------
  Se valor_absoluto(x) = y, então:
    - Ou x >= 0 e y =  x ,
    - Ou x <  0 e y = -x .
  ----------------------------------------
```

Na linguagem de especificação ACSL, isso pode ser escrito assim (no código-fonte essas linhas ficam logo acima do código da função "valor_absoluto"):

```
  ------------------------------------------------
  /*@
      ensures x <  0 ==> \result == - \old(x) ;
      ensures x >= 0 ==> \result ==   \old(x) ;
   */
  ------------------------------------------------
```

Observações:

- A especificação em ACSL é fornecida dentro de um comentário no código-fonte, iniciado por "/*@".

- Uma cláusula "ensures" descreve uma PÓS-CONDIÇÃO da função, isto é, uma garantia sobre o cálculo por ela realizado. A cláusula é TERMINADA POR PONTO-E-VÍRGULA.

- O termo "\result" denota o valor retornado pela função.

- O termo "\old(...)" se refere ao valor de uma variável no início da chamada da função; sem isso, uma ocorrência de uma variável numa pós-condição denota o valor da variável ao final da execução da função.

Suponha que a especificação ACSL e a função "valor_absoluto" acima estão gravados num arquivo "codigo.c"; nesse caso, nós podemos verificá-lo por meio do plug-in WP da plataforma Frama-C executando num terminal:

```
frama-c-gui -wp codigo.c &>/dev/null &
```

Uma janela se abrirá; à direita, estará o código-fonte original; à esquerda dele estará uma versão equivalente mas ligeiramente modificada dele, na forma esperada pela plataforma.

Clicando-se com o botão direito do "mouse" em uma pós-condição e então na opção "Prove property by WP", aparecerá uma bola verde à esquerda da pós-condição, indicando que o código foi verificado com sucesso.

1. EXERCÍCIOS:

   a) Experimente verificar o código acima na prática, da maneira descrita.

   b) Escreva em C uma função que recebe dois inteiros e retorna o maior deles. Escreva em ACSL uma especificação para essa função, e então verifique o código via Frama-C/WP.

UM EXEMPLO COM LAÇOS

Aqui está uma versão em C/ACSL do algoritmo de busca linear:

```
  ============================================================================
  /*@
    @  requires n >= 0;
    @
    @  ensures   0 <= \result && \result <= \old(n) ;
    @
    @  ensures ( 0 <= \result && \result < \old(n) )
    @          ==> (
    @               \old(v) == \old(A[\result])
    @               &&
    @               \forall integer i;
    @                 ( 0 <= i && i < \result ) ==> \old(v) != \old(A[i])
    @              ) ;
    @
    @  ensures ( \result == \old(n) )
    @          ==> (
    @               \forall integer i;
    @                 ( 0 <= i && i < \old(n) ) ==> \old(v) != \old(A[i])
    @              ) ;
    @
    @  assigns \nothing ;
    */

  int busca_linear (int *A, int n, int v)
    {
    int i = 0;
    /*@
      @  loop invariant 0 <= i && i <= n ;
      @  loop invariant \forall integer j;
      @                 (0 <= j && j < i) ==> (v != A[j]) ;
      @  loop assigns i ;
      */
    while (i < n) {
      if (A[i] == v)
        { return i; }
      ++i;
      } // END while
    return i;
    } // END busca_linear
  ============================================================================
```

Observações:

- Apenas a primeira "@" de cada comentário-especificação é necessária.

- Uma cláusula "requires" especifica uma pré-condição, isto é, requisitos para que a função funcione corretamente.

- Uma cláusula "assigns" lista aquilo que é modificado pelo código. Acima, a cláusula "loop assigns i" garante que apenas a variável "i" é modificada durante o laço, e a cláusula "assigns \nothing" garante que a função não modifica nada exceto suas variáveis locais.

2. EXERCÍCIOS:

   a) Verifique a função acima (via Frama-C/WP). Observe que as pós-condições
      não serão provadas até que os invariantes estejam provados.

   b) Escreva e verifique uma função em C que retorna o maior elemento de um
      vetor. (Use "\exists" assim como "\forall".)

   c) Escreva e verifique uma função de busca binária em C.

 INDO ALÉM

O tutorial indicado nas referências abaixo mostra informações adicionais interessantes, como a possibilidade de usar o plug-in RTE para provar a ausência de erros de execução em um programa: acesso a elemento inválido de vetor, etc.

A especificação da ACSL fornece mais detalhes sobre a sintaxe das anotações. Uma boa maneira de aprender é escrever e verificar funções e recorrer à especificação da linguagem quando necessário. Além disso, o "ACSL by example" fornece numerosos exemplos de uso da linguagem.

Caso você precise de ajuda especializada, você fazer uma pergunta em

  http://stackoverflow.com/tags/frama-c/ ,

mas se certifique de estudar os recursos à disposição antes de fazer perguntas.

 REFERÊNCIAS

O seguinte tutorial é uma excelente introdução, de 1 aula, ao uso do Frama-C:

https://verify.iaik.tugraz.at/teaching/vt/pub/Main/AssignmentThree2013/frama_tutorial.pdf

O sítio da plataforma é rico em informações sobre ela, e possui (em "Download") manuais sobre a ferramenta, os plug-ins e a linguagem de anotação ACSL:

  http://frama-c.com/

Este material fornece muitos exemplos de programas verificados via Frama-C:

  http://www.fokus.fraunhofer.de/download/acsl_by_example