Universidade de Brasília - Instituto de Ciências Exatas
Departamento de Ciência da Computação

117366 - Lógica Computacional 1

Turmas A e D - 2015/1

Descrição do Projeto

Formalização do Algoritmo de Ordenação pelo Método da Bolha (*BubbleSort*)

14 de maio de 2015

Prof. Mauricio Ayala-Rincón Prof. Flávio L. C. de Moura

O estagiário de docência Lucas Ângelo da Silveira (`lucas.angel9@gmail.com`) dará suporte aos alunos no desenvolvimento do projeto. Laboratórios do LINF têm instalado o *software* necessário (PVS 6.0 com as bibliotecas PVS da NASA).

1 Introdução

Algoritmos de busca e ordenação são fundamentais em Ciência da Computação. Neste projeto considerar-se-ão algoritmos de ordenção sobre o tipo abstrato de dados `list` como especificado no assistente de demonstração PVS.

O objetivo do projeto da disciplina é introduzir os mecanismos básicos de manuseio de tecnologias de verificação e formalização que utilizam técnicas dedutivas lógicas, como as estudadas na disciplina, para garantir a correção lógica de objetos computacionais.

2 Descrição do Projeto

Com base na *teoria* `sorting`, composta de formalizações para auxiliar na prova de correção de diferentes algoritmos de ordenação e provas auxiliares, este projeto trata da especificação e da prova da correção do algoritmo de ordenação pelo método da bolha, que são, respectivamente, apresentadas nos arquivos `bubblesort.pvs` (especificação) e `bubblesort.prf` (formalização). Os arquivos estão disponíveis na página da disciplina, O algoritmo está especificado na linguagem do assistente de demonstração PVS (`pvs.csl.sri.com`), executável em plataformas Unix/Linux/OS. Os alunos deverão formalizar propriedades de correção das especificações para ordenação pelo método da bolha sobre listas de naturais.

2.1 O Método da Bolha: *Bubblesort*

Diversas noções e lemas auxiliares, demonstrados integralmente na teoria `sorting`, são necessários. Esses elementos são a base para o desenvolvimento de uma formalização da correção de diversos algoritmos de ordenação.

Em particular, neste projeto, o objetivo é demonstrar, formalmente, que a especificação de ordenação pelo método da bolha é correta:

```
bubblesort(l) : list[nat] =
IF null?(l) THEN l
ELSE bubblesort_aux(l, l`length - 1 )
ENDIF
```

Nessa especificação, o mecanismo de ordenação se dá ao deslocar (borbulhar) o maior elemento da lista para a última posição (`l`length -1`) da sublista não ordenada; logo, o segundo maior elemento para a sublista até a penúltima posição (`l`length -2`), e assim recursivamente. Este deslocamento por borbulhamento ou flutuação é aplicado via a função auxiliar `bubblesort_aux` definida como a seguir:

```
bubblesort_aux(l, ( n : below[l`length ] )) : RECURSIVE list[nat] =
IF n = 0 THEN l
ELSE bubblesort_aux(bubbling(l, n), n-1)
ENDIF
MEASURE n
```

A função recursiva `bubblesort_aux` recebe como argumentos uma lista `l` e um natural `n`, que deve ser estritamente menor do que o comprimento da lista `l`. O parâmetro `n` tem como objetivo controlar o número de vezes que o borbulhamento deve ser executado. A função `bubbling` é responsável por fazer o borbulhamento dos primeiros `n` elementos da lista `l`, isto é, `bubbling(l, n)` vai deslocar o maior elemento da sublista que contém os primeiros `n` elementos da lista `l` para a `n`-ésima posição desta sublista:

```
bubbling(l, ( n : below[ l`length])) : RECURSIVE  list[nat] =
IF n = 0 THEN l
ELSIF car(l) > car(cdr(l)) THEN
cons(car(cdr(l)), bubbling(cons(car(l),cdr(cdr(l))), n - 1))
ELSE cons(car(l), bubbling(cdr(l), n - 1))
ENDIF
MEASURE n
```

3 Questões

As questões a serem resolvidas aparecem no arquivo `bubblesort.pvs` como conjecturas (`CONJECTURE`). Todos os outros resultados estão provados[1].

1. Prove que a função `bubblesort_aux` preserva o tamanho da lista recebida como entrada. A prova é por indução em `n` (`measure-induct+ n (l n)`):

   ```
   bubblesort_aux_preserves_length : CONJECTURE
   FORALL  (l: list[nat], n :below[l`length]) :
     length(bubblesort_aux(l,n)) = length(l)
   ```

[1] Veja o status geral da teoria com o comando `M-x prove-importchain`. O status de um arquivo pode ser visto com `M-x prove-theory`

2. Prove que o resultado de se aplicar a função `bubbling` a uma lista `l` e natural `n` menor do que o comprimento da lista `l`, retorna uma lista que é uma permutação de `l`. Para provar este lema você precisará de alguns lemas sobre transitividade, `cons` e eliminação de elementos de uma permutação (disponíveis na teoria `sorting`). A prova é por indução em `n` (`measure-induct+ n (l n)`):

```
bubbling_preseves_contents :  CONJECTURE
FORALL (l: list[nat], n :below[l‘length]) :
 permutations(l, bubbling(l,n))
```

3. Prove que a função `bubbling(l,n)` move o maior elemento da sublista contendo os primeiros `n` elementos de `l` (`n`-prefixo de `l`), e o coloca na `n`-ésima posição de `l`. Este é o teorema mais relevante no que se refere a função `bubbling`, e deve ser provado por indução como na questo anterior.

```
bubbling_bubbles : CONJECTURE
FORALL  (l: list[nat], n :below[l‘length]) :
 LET l1 = bubbling(l,n) IN
   FORALL (i:nat) :  i <= n IMPLIES nth(l1,i) <= nth(l1,n)
```

4. Prove que a função `bubblesort_aux` preserva o `n`-prefixo da lista original `l` no sentido que os `n`-prefixos (da lista original e de `bubblesort_aux(l,n)`) são permutações um do outro, isto é, para cada elemento no `n`-prefixo de `bubblesort_aux(l,n)` existe uma posição no `n`-prefixo de `l` contendo o mesmo elemento. Utilize lemas anteriores e da teoria `sorting` como, por exemplo, `bubblesort_aux_preserves_suffix`, `permutations_of_app_pref`, `permutations_preserves_contents`, `contents_prefix`, `length_prefix`, `length_suffix`, etc.

```
bubblesort_aux_preserves_prefix : CONJECTURE
FORALL  (l: list[nat], n :below[l‘length]) :
 LET l1 = bubblesort_aux(l,n) IN
   FORALL (i:below[n+1]) :  EXISTS (j:below[n+1]) : nth(l1,i) = nth(l,j)
```

4 Etapas do desenvolvimento do projeto

Os alunos deverão definir os grupos de trabalho limitados a **três** membros até o dia **04.05.2015**. Exceto pelo dia da segunda prova, **01.07.2015**, as aulas serão realizadas no LINF a partir do dia **04.05.2015**.

O projeto será dividido em duas etapas como segue:

- A primeira etapa do projeto é a de Verificação das Formalizações. Os grupos deverão ter prontas as suas formalizações na linguagem do assistente de demonstração PVS e enviar por e-mail ao estagiário com cópia para o professor os arquivos de especificação e de provas desenvolvidos (`bubblesort.pvs` e `bubblesort.prf`) até o fim da manhã do dia **08.06.2015**. Nos dias **8 e 10.06.2015**, durante o horário de aula, realizar-se-á a verificação do trabalho para a qual os grupos deverão, em acordo com o monitor e professor, determinar um horário (de mánimo uma hora) no qual todos membros do grupo deverão comparecer.
  Avaliação (peso 6.0):

  - Um dos membros, selecionado por sorteio, explicará os detalhes da formalização em máximo 20 minutos.
  - Os quatro membros do grupo poderão complementar a explicação inicial em máximo 10 minutos.
  - A formalização poderá ser então testada em máximo 30 minutos.

- A segunda etapa do projeto consiste da apresentação dos resultados finais e conclusões do estudo do problema.
  Avaliação (peso 4.0): Cada grupo de trabalho devera entregar um Relatório Final inédito, editado em LaTeX, limitado a oito páginas (12 pts, A4, espaçamento simples) do projeto até o dia **22.06.2015** com o seguinte conteúdo:

  - Introdução e contextualização do problema.
  - Explicação da soluções.
  - Especificação do problema e explicação do método de solução.
  - Descrição da formalização.
  - Conclusões.
  - Referências.