Tarefa 01 – 2014/2
Tema: Conjuntos Indutivos, Funções Recursivas e Provas por Indução
Disciplina: Métodos Formais para Computação
Professor: Alfio Martini

1 Objetivos

Trabalhar com definições de funções primitivas recursivas e provar alguns teoremas sobre as propriedades destas funções. Como bibliografia sobre o assunto, sugerimos [4, 8, 5, 7, 2, 3, 6, 9, 1].

2 Sobre a Especificação do Trabalho

2.1 Primeira opção

Definir uma teoria para árvores binárias polimórficas (genéricas), com operações para:

① enumerar os nodos da árvore após visitála na forma *inorder*;

② enumerar os nodos da árvore após visitála na forma *postorder*;

③ enumerar os nodos da árvore após visitála na forma *preorder*;

④ refletir (ou espelhar) uma árvore

Note que as primeiras três funções acima retornam uma lista com os nodos visitados na ordem especificada.

Com as opções acima você deve demostrar os seguintes teoremas:

① Refletir uma árvore e após visitá-la na forma postorder é a mesma coisa que listar os nodos após uma visita em preorder e depois reverter a enumeração correspondente.

② Refletir uma árvore, após visitá-la na forma inorder e inverter a enumeração correspondente é a mesma coisa que listar os nodos visitados na forma inorder.

Ambas as provas acima devem ser feitas de forma detalhada tanto manualmente como na linguagem de prova Isar. Os lemas (teoremas auxiliares) necessários podem ser assumidos. Estes teoremas serão necessários na prova detalhada da linguagem Isar. Neste caso, eles podem ser demonstrados por automação com a linguagem script dos comandos apply.

2.2 Segunda opção

Aqui você deve definir um tipo de dado (recursivo) para expressões aritméticas (sobre naturais ou inteiros). Estas expressões devem conter, pelo menos:

- variáveis;
- constantes;

- somas e
- multiplicações.

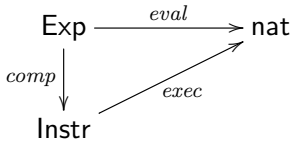
Estas expressões compõem o que chamamos aqui da nossa linguagem de alto nível. Chamaremos essa linguagem de Exp. Além disso, queremos compilar essas expressões para uma máquina de pilha que contém instruções para:

- carregar uma constante na pilha;
- carregar um valor na pilha;
- somar dois valores que estejam topo de uma pilha;
- multiplicar dois valores que estejam no topo da pilha

A linguagem da pilha também é um tipo de dado. Chamaremos essa linguagem de Instr. Uma pilha pode ser modelada como uma lista, onde a cabeça da lista é o topo da pilha e a cauda da lista é o resto da pilha. Com esses dois tipos de dados, é necessário agora definir algumas funções:

- uma função *eval* que recebe uma expressão aritmética, uma função que mapeia variáveis para valores e retorna o número que denota a expressão.
- uma função *compile* que traduz expressões aritméticas para a linguagem da pilha.
- uma função *exec* que recebe a linguagem compilada e retorna o valor da expressão

Agora, o teorema que precisa ser provado é simples: *o valor da expressão de alto nível tem que ser igual ao valor da expressão compilada, ou seja o triângulo abaixo comuta (isto é, os dois caminhos de* Exp *para* nat *sao iguais:*



Para ter uma idéia de como modelar tais expressões, ver por exemplo, [7], seção 2.5.6, onde um exemplo com uma linguagem de para expressões booleanas é trabalhado em grande detalhe.

A prova do teorema acima deve ser feita de forma detalhada tanto manualmente como na linguagem de prova Isar. As provas dos lemas (teoremas auxiliares) não são necessários. Podem ser assumidos. Por outro lado, estes teoremas serão necessários na prova detalhada da linguagem Isar. Neste caso, eles podem ser demonstrados por automação com a linguagem script dos comandos apply.

3 Sobre o Artigo da Tarefa

Na apresentação da tarefa, deve deve ser utilizado o modelo da SBC (Sociedade Brasileira da Computação) para a construção do artigo. Neste artigo, além das definições e demonstrações colocadas acima, você deve incluir:

1. Uma fundamentação teórica sobre o assunto tratado nesta tarefa.

2. A apresentação das definições, dos teoremas e suas demonstrações como colocado acima.

3. Uma discussão detalhada comparando a estrutura de uma prova manual e sua implementação correspondente em Isar. Esta discussão comparativa deve incluir, pelo menos:

   - a declaração de variáveis arbitrárias;
   - a declaração das hipóteses, da tese da prova e da tese das várias subprovas;
   - a computação detalhada das igualdades (via simplificação).

4. Bibliografia utilizada.

4 Sobre a Pontuação do Trabalho

A seguinte pontuação será utilizada para avaliar a tarefa:

Implementação	
`Ítem`	`Pontuação`
Provas	6 pontos
Discussão	2 pontos
Apresentação	2 pontos

5 Sobre a Entrega do Trabalho

① A tarefa deve ser realizada individualmente.

② Na submissão da tarefa, enviar um arquivo zipado, nomeado por exemplo, como `NomeSobrenome`, o qual inclui:

   - o artigo no formato `pdf`;
   - os arquivos `*.thy`, contendo as definições dos tipos, das funções recursivas e as provas passo-a-passo na linguagem Isar.

6 Sobre a Bibliografia

Neste documento estão incluídos livros, artigos e manuais de referência que podem ser utilizados como bibliografia na tarefa. Para esta tarefa, não referenciar sites e páginas da Internet.

Referências

[1] Alfio Martini. Verificação Formal em Isabelle/Isar para Newbies. Notas de aula da disciplina de Métodos Formais, 2012.

[2] Robin Milner Michael Gordon and Christopher Wadsworth. Edinburgh LCF - A Mechanized Logic for Computation. In *Lectute Notes in Computer Science, vol. 78*. Springer-Verlag, 1980.

[3] Tobias Nipkow. A Tutorial Introduction to Structured Isar Proofs. 2011.

[4] L.C. Paulson. *ML for the Working Programmer*. Cambridge University Press, 1997.

[5] P. Rudnicki. An Overview of the Mizar Project. In *Workshop on Types for Proofs and Programs*, 1992.

[6] Jeremi Siek. Practical Theorem Proving with Isabelle/Isar. Lecture Notes, 2007.

[7] Lawrence Paulson Tobias Nipkow and Markus Wenzel. Isabelle/HOL – A Proof Assistant for Higher-Order Logic. In *Lectute Notes in Computer Science, vol. 2283*. Springer-Verlag, 2002.

[8] Markus Wenzel. *Isabelle/Isar – a versatile environment for human-readable formal proof documents*. PhD thesis, Institut fuer Informatik, Technische Universitaet Muenchen, 2002.

[9] Markus Wenzel. The Isabelle/Isar Reference Manual. Documentation available at the Isabelle website, 2012.