Universidade de Évora

Linguagens de Programação

Relatório do 2º trabalho prático

Autores:
Bernardo Pinheiro, 29205
Luis Teimas, 31561
Beatriz Alvito 32436

Docentes:
Teresa Gonçalves

Maio de 2017

Índice

1 Introdução

No âmbito da unidade curricular de Linguagem de Progamação, na realização do segundo trabalho prático, pretende-se continuar com a abordagem desenvolvida no primeiro trabalho, para que este, após a mudança das variáveis (no caso de ser necessário) faça a redução, se possível.

2 Desenvolvimento

2.1 Alterações realizadas

Para a realização do trabalho, tivemos de alterar o primeiro, face ao problema encontrado. Achamos que seria mais simples mudar o funcionamento da nossa árvore, enquanto que no primeiro tínhamos listas de listas, neste alterámos de forma a ter uma arvóre convencional. Sendo esta uma árvore com nós, tal e qual como faríamos no papel, cada nó tem um valor, um tipo, um nó á esquerda e um á direita.

2.2 Funcionamento

O trabalho é constituído por 5 ficheiros:

- makefile: contém as informações necessárias para correr o trabalho, através do comando make.
- lexer.py: faz a análise lexical, contendo os tokens necessários para o funcionamento do programa, o seu "significado"para o interpretador, bem como as precedências necessárias.
- parser.py: faz a análise sintática, contém a gramática do programa e a forma como esta se comporta, contém também a inicialização das listas que são utilizadas no ficheiro lambda.py
- tree.py: contém a estrutura da árvore.
- lambda.py: este ficheiro contém várias funções que constituem o funcionamento do programa, sendo elas:
  - random_alfa(alfa): Devolve a primeira letra da lista alfa, que contém todas as letras do alfabeto.
  - createTree(): Como o nome indica, cria a árvore.
  - insertNodes(i, currentnode): Esta função ajuda na criação da árvore, usando a raíz da mesma para ir inserindo nós à esquerda e à direita recursivamente.
  - checkRep(free): Verifica se existem repetições de variáveis(ligadas e livres) e substitui essas variáveis se necessario.
  - freeVar(): Devolve uma lista com todas as variáveis livres, se elas existirem, removendo todas as variáveis ligadas de uma lista de todas as variáveis da expressão.
  - call(rootnode): Verifica se é necessário fazer uma redução.
  - **callByValue(node): Implementa a redução através da estrategia call-by-value.**
  - countNodes(node, a): Faz a contagem do número de variáveis que é necessário mudar.
  - replaceNode(node, newnode): Substitui um nó por um novo nó
  - replaceNodeRec(node, a, newnode): Substitui todos os nós que tenham como valor a por um novo nó.

- updateRep(node) e uptRep(node): Actualiza as listas de repetições e de variáveis para poder realizar mudanças das mesma, no caso de haver conflito.
- run(): Corre o programa.

2.3 Exemplos

```
Input: (!x.(!y.y)(xx))(!z.z)b

<-(!x.x((!y.y)x))(!z.z)b
->(!a.a((!y.y)a))(!z.z)b
->b

Input: (!x.x)(!x.x)(!x.x)(!x.x)

<-(!x.x)(!x.x)(!x.x)(!x.x)
->(!a.a)(!b.b)(!c.c)(!d.d)
->(!d.d)

Input: (!x.xxx)(!y.y)

<-(!x.xxx)(!y.y)
->(!x.xxx)(!y.y)
->(!c.c)

Input: (!x.(!y.y)x)b

<-(!x.(!y.y)x)b
->(!a.(!y.y)a)b
->b

Input: (!z.zz)(!y.y(!x.x)b)
<-(!z.zz)(!y.y(!x.x)b)
->(!z.zz)(!y.y(!x.x)b)
->bb
```

Em relação aos exemplos apresentados, o primeiro não funciona correctamente. Este problema ocorre pois quando existe uma ou mais variáveis entre parênteses que não estão "directamente ligadas"o programa não as reconhece devidamente.
Este caso (xx) é o que vai fazer com que o funcionamento não seja correcto.

3 Conclusão

Em relação ao trabalho final, estamos satisfeitos com o resultado, uma vez que conseguimos respeitar o enunciado, fazendo uma aproximação ao mesmo eficiente. No entanto, em casos em que o exemplo acima referido aconteça, o programa não vai funcionar devidamente. Este problema reflecte-se também no exemplo 10 do enunciado, neste caso serão (bc),(ba) e (cd) os causadores do problema.
Foi imensamente produtivo em diversos aspectos a elaboração deste projecto, tanto técnicamente como em desenvolvimento de capacidade de trabalhar em grupo.

4 Bibliografia

A lista de consulta para este trabalho não foi alterada, sendo ela:

- http://www.inf.fu-berlin.de/lehre/WS03/alpi/lambda.pdf
- http://www.cs.cornell.edu/courses/cs3110/2008fa/recitations/rec26.html
- http://pages.cs.wisc.edu/~horwitz/CS704-NOTES/1.LAMBDA-CALCULUS.html
- http://www.dabeaz.com/ply/PLYTalk.pdf
- http://www.dabeaz.com/ply/ply.html