Lobisomen

Caroline Rosa e Henrique Fontenelle

26/04/2019

Universidade Federal Fluminense

Linha principal

- automaton.auto(parse.generator(entrada))

- Sendo entrada uma String pega pelo conteudo do documento "entrada.txt"

Divisão do Projeto

- ParseLpeg.lua - Lexer, Parser, Transformer
- automaton.lua - Automaton
- localization.lua - Localization
- entrada.txt - Entrada do codigo
- lobisomen.lua - Main

ParseLpeg.lua

Recebe um código Imp e realiza:

- Lexit
- Parser
- Transformer

Gramática LPeg

•Definição básica da gramática:

$$\text{integer} = \text{white} * \text{lpeg.R("09")}^{1} / tonumber$$
$$localmultDivValue = white * lpeg.C(lpeg.S("/*"))$$
$$andOr = white * (lpeg.C("and") + lpeg.C("or"))$$
$$notValue = white * lpeg.C("Not")$$

Generator

- Como é a execução do programa para o comando: bola := 3 + 2?

$$
\begin{aligned}
&\text{"s"},\\
&\text{s} = \text{impFinal}(\text{lpeg.V}(\text{"cmd"})^1),\\
&cmd = lpeg.V(\text{"}assign\text{"}) + lpeg.V(\text{"}loop\text{"}) + lpeg.V(\text{"}cond\text{"}) + lpeg.V(\text{"}exp\text{"}),\\
&assign = node(lpeg.V(\text{"}id\text{"}) * igual * lpeg.V(\text{"}exp\text{"})),\\
&id = node(idValue),\\
&exp = (lpeg.V(\text{"}boolExp\text{"}) + lpeg.V(\text{"}aritExp\text{"})),\\
&aritExp = lpeg.V(\text{"}addSub\text{"}) + lpeg.V(\text{"}multExp\text{"}) + lpeg.V(\text{"}multDiv\text{"}),\\
&addSub = node(lpeg.V(\text{"}multExp\text{"}) * addSubValue * lpeg.V(\text{"}aritExp\text{"})),\\
&multExp = lpeg.V(\text{"}multDiv\text{"}) + lpeg.V(\text{"}atom\text{"}) + lpeg.V(\text{"}parentesisExp\text{"}),\\
&atom = node(integer + lpeg.V(\text{"}bool\text{"}) + lpeg.V(\text{"}id\text{"}))
\end{aligned}
$$

Função node()

- Responsável por realizar o parser/transformer a nível de comando

```
local function node(p)
   return p / function(left, op, right,...)
      op = transformType[op]()
      return {op, left, right }
  end
end
```

- De acordo com o token, realiza o parser/transformer correspondente na lista de funções transformType

```
transformType =
  {
    ["+"]=typeSum,
    [-"]=typeSub,
    ["*"]=typeMul,...
   }
```

Função tranformType()

- Exemplo de um tranformType

```
function typeSum()
  return "SUM"
end
```

Lexit

- Antes de finalizar a recursão, a função impFinal() é realizada

```
local function impFinal(p)
  return p / function(...)
   local arg={...}
    local resp;
     if( #arg ≥ 2) then
      atual = #arg
      resp = "CSEQ",arg[atual-1], arg[atual]
      atual = atual -1
      while atual  ≥ 1 do
        resp = "CSEQ",arg[atual], resp
        atual = atual -1
      end
    else
      resp = arg[1]
    end
  return resp
end
```

Automaton.lua

- Recebe a arvore montada pelo PaserLpeg.generator

- Inicia a pilha de controle, pilha de valores, o enviroment e o storage

- Comeca a sua funcao recursiva automaton.rec, que para quando a pilha de controle esta vazia

Automaton.rec

- Sempre pega o head da pilha de controle e olha seu primeiro index
- Este index diz como ele deve tratar aquele "conjunto"(operacoes,comandos,numeros,etc)
- De acordo com este valor ele envia para um handler especifico que executa o que foi descrito na documentacao π in a nutshell, como se executa o δ deste valor.
- Ex : $\delta(Id(W) :: C, V, E, S) = \delta(C, B :: V, E, S), whereE[W] = I \wedge S[I] = B$
- Por fim ocorre outra chamada, ate que a pilhade controle esteja vazia logo apos uma chamada

δ e OCs

- A documentacao aponta o empilhamento de "OCs" no δ de varias expressoes e comandos
- Para estas possuimos os handlers qeu apenas empilham os "operacao", e os que o executam em si
- EX: handle_operacao $\rightarrow$ empilha #operacao, o qual possui o handle_h_operacao

Enviroment e Storage

•Ao usarem ASSING precisamos que seja criada uma ligacao com o ID, o enviroment e o storage

•O enviroment deve conter um "localization"que aponte onde do Storage esta o valor ligado ao ID que sofreu ASSING

•Como em Lua uma string pode ser usada como um index de uma table, usamos o ID como o indexador do da table do enviroment, que no determinado local terah o LOC apontando para o Storage

•Ex: env['ID'] = {LOC,x}
str[x] = ' valor ligado ao ID '

Location

- Consiste de um valor SIZE e a funcao location.getLoc, a qual atualizada SIZE como SIZE + 1 e o restorna
- A funcao .getLoc eh chamada pelos ASSING para saber onde armazenar na STORAGE a informacao de um ID

getValue

- Funcao especial para o tratamento daqules que possuim a classificacao : NUM, BOO, ID, LOC
- Feita de forma retornar valores que possam ser operados por aqueles que a chamou

Finalizacao

Quando a pilha de controle eh encontrada vazia no inicio da execucao da funcao recursiva

- Limpa o Head para beleza de impressao
- Imprime todo o conteudo do automato no momento que terminou
- Retorna pilha de valores e o storage (uma vez que em teoria apenas esses que deveriam continuar afetador ao final da execucao de um programa)