Universidade Estadual de Santa Cruz – UESC

Relatórios de Implementações de Analisador Descendente Recursivo para GLC

Docente César Alberto Bravo Pariente

Discente Matheus Miranda Brandão

Matrícula 201820065

Disciplina Compiladores.

Curso Ciência da Computação

Semestre 2022.2

Ilhéus – BA
2022

Índice

GLC LL

O projeto consiste na implementação em C de um algoritmo que busca simular um Analisador Descendente Recursivo que reconhece a linguagem gerada por uma gramática livre de contexto. A execução do código recebe como entrada a palavra e retorna se a palavra foi aceita ou ocorreu um erro, suas produções e o vetor referente a Árvore de Análise.

O analisador implementado ignora o token ' '.

Produções:

p_1: S -> M | GM | NGM

p_4: N -> n(){ C; r(E); }

p_5: G -> g(){ C; r(E); }

p_6: M -> m() { C; r(E); }

p_7: E -> 0 | 1 | x | y | (EXE)

p_{12}: X -> + | - | * | /

p_{16}: C -> h=E | i=E | j=E | k=E | z=E | (EXE) | w(E){ C; } | f(E){ C; } | o(E;E;E){ C; }

Compilando e Executando

Para a execução não é necessário o uso de nenhuma dependência, basta compila-lo normalmente.

$ gcc proj2_b.c -o proj2_b

Ao executar é necessário digitar o nome do arquivo destino contendo as palavras, caso contrário resultará em erro.

Exemplo:

$ ./proj2_b examples/inputs.txt

Neste projeto pode-se adicionar num .txt todas as palavras separadas por uma quebra de linha. Em caso de erro, pularemos para a próxima palavra.

Exemplo:

```
m(){ r(1); }
m(){ h=(x+y); r(0); }
m(){ (1-1); r(1); }
m(){ w(1) { (1/x); }; r(1); }
n() { (0/y); r(y); } g() { i=y; r(x); } m() { (1-x); r(0); }
m(){w(x){f(y){k=(1+(1*0));};};r(0);}
```

Testes

Caso a entrada dada seja incorreta o programa irá imprimir os tokens até o momento, então avisará sobre o erro, informará qual o código e o token inesperado, então pulará para a próxima palavra. Como outputs temos a palavra analisada, as produções e a árvore sintática (no formato '[index | mapeamento | token]'), foi considerado o pior caso, onde temos uma árvore n-ária de 12. A árvore foi feita seguindo o modelo de árvore compacta.

Para criação de palavras compatíveis com a linguagem gerada pela GLC foi utilizado o website "CFG Developer".

m(){ r(1); }

Palavra 1: m(){

Erro 83. Token 'r' inesperado. Prosseguindo para proxima palavra.

Producoes: P1, P6

Arvore: [0|0| S], [1|1| M], [2|13| m], [3|14| ( ], [4|15|)], [5|16| {], [6|17| C], [7|18| ;],

[8|19| r], [9|20| ( ], [10|21| E], [11|22|)], [12|23| ;], [13|24| }]

m(){ h=(x+y); r(0); }

Palavra 2: m(){ h=(x+y); r(0); }

Palavra aceita.

Producoes: P1, P6, P16, P11, P9, P12, P10, P7

Arvore: [0|0| S], [1|1| M], [2|13| m], [3|14| ( ], [4|15|)], [5|16| {], [6|17| C], [7|18| ;],

[8|19| r], [9|20| ( ], [10|21| E], [11|22|)], [12|23| ;], [13|24| }], [14|205| h],

[15|206| =], [16|207| E], [17|2485| ( ], [18|2486| E], [19|2487| X], [20|2488| E],

[21|2489|)], [22|29833| x], [23|29845| +], [24|29857| y], [25|253| 0]

m(){ (1-1); r(1); }

Palavra 3: m(){ (1-1); r(1); }

Palavra aceita.

Producoes: P1, P6, P21, P8, P13, P8, P8

Arvore: [0|0| S], [1|1| M], [2|13| m], [3|14| ( ], [4|15|)], [5|16| {], [6|17| C], [7|18| ;], [8|19| r], [9|20| ( ], [10|21| E], [11|22|)], [12|23| ;], [13|24| }], [14|205| ( ], [15|206| E], [16|207| X], [17|208| E], [18|209|)], [19|2473| 1], [20|2485| -], [21|2497| 1], [22|253| 1]

m(){ w(1) { (1/x); }; r(1); }

Palavra 4: m(){ w(1) { (1/x); }; r(1); }

Palavra aceita.

Producoes: P1, P6, P22, P8, P21, P8, P15, P9, P8

Arvore: [0|0| S], [1|1| M], [2|13| m], [3|14| ( ], [4|15|)], [5|16| {], [6|17| C], [7|18| ;], [8|19| r], [9|20| ( ], [10|21| E], [11|22|)], [12|23| ;], [13|24| }], [14|205| w], [15|206| ( ], [16|207| E], [17|208|)], [18|209| {], [19|210| C], [20|211| ;], [21|212| }], [22|2485| 1], [23|2521| ( ], [24|2522| E], [25|2523| X], [26|2524| E], [27|2525|)], [28|30265| 1], [29|30277| /], [30|30289| x], [31|253| 1]

n() { (0/y);r(y); } g() { i=y;r(x); } m() { (1-x);r(0); }

Palavra 5: n() { (0/y); r(y); } g() { i=y; r(x); } m() { (1-x); r(0); }

Palavra aceita.

Producoes: P3, P4, P21, P7, P15, P10, P10, P5, P17, P10, P9, P6, P21, P8, P13, P9, P7

Arvore: [0|0| S], [1|1| N], [2|2| G], [3|3| M], [4|13| n], [5|14| ( ], [6|15|)], [7|16| {], [8|17| C], [9|18| ;], [10|19| r], [11|20| ( ], [12|21| E], [13|22|)], [14|23| ;], [15|24| }], [16|205| ( ], [17|206| E], [18|207| X], [19|208| E], [20|209|)], [21|2473| 0], [22|2485| /], [23|2497| y], [24|253| y], [25|25| g], [26|26| ( ], [27|27|)], [28|28| {], [29|29| C], [30|30| ;], [31|31| r], [32|32| ( ], [33|33| E], [34|34|)], [35|35| ;], [36|36| }], [37|349| i], [38|350| =], [39|351| E], [40|4213| y], [41|397| x], [42|37| m], [43|38| ( ], [44|39|)], [45|40| {], [46|41| C], [47|42| ;], [48|43| r], [49|44| ( ], [50|45| E], [51|46|)], [52|47| ;], [53|48| }], [54|493| ( ], [55|494| E], [56|495| X], [57|496| E], [58|497|)], [59|5929| 1], [60|5941| -], [61|5953| x], [62|541| 0]

m(){w(x){f(y){k=(1+(1*0));};};r(0);}

Palavra 6: m(){w(x){f(y){k=(1+(1*0));};};r(0);}

Palavra aceita.

Producoes: P1, P6, P22, P9, P23, P10, P19, P11, P8, P12, P11, P8, P14, P7, P7

Arvore: [0|0| S], [1|1| M], [2|13| m], [3|14| ( ], [4|15|)], [5|16| {], [6|17| C], [7|18| ;], [8|19| r], [9|20| ( ], [10|21| E], [11|22|)], [12|23| ;], [13|24| }], [14|205| w], [15|206| ( ], [16|207| E], [17|208|)], [18|209| {], [19|210| C], [20|211| ;], [21|212| }], [22|2485| x], [23|2521| f], [24|2522| ( ], [25|2523| E], [26|2524|)], [27|2525| {], [28|2526| C], [29|2527| ;], [30|2528| }], [31|30277| y], [32|30313| k], [33|30314| =], [34|30315| E], [35|363781| ( ], [36|363782| E], [37|363783| X], [38|363784| E], [39|363785|)], [40|4365385| 1], [41|4365397| +], [42|4365409| ( ], [43|4365410| E], [44|4365411| X], [45|4365412| E], [46|4365413|)], [47|52384921| 1], [48|52384933| *], [49|52384945| 0], [50|253| 0]

Link para download

Código fonte e exemplos encontram-se para download no seguinte link:

https://github.com/MatBrands/Compiladores/tree/master/Proj2/Proj2b%20Compactado

Referências

https://web.stanford.edu/class/archive/cs/cs103/cs103.1156/tools/cfg/