LR(0) Parser documentation

Team:

Mihalache Stefan Cristian
Mos Daniele

Github repo: https://github.com/StefanCsPurge/Formal-Languages-and-Compiler-Design

Parser execution flow:

Instatantiate Grammar class ->
instantiate Parser class ->
read sequence from file ->
call parser.parseSequence(sequence, outFile)

Most important modules:

- FiniteAutomata
- Parser
- ParserOutput
- ST (Symbol Table)
- Scanner
- TableEntry
- TreeNode (for the Parsing Tree)

Finite Automata

Class that contains all the data needed for the Finite Automata: Q (all states), E (alphabet), q0 (initial state), F (final state), Tr (transitions). There's also a method to read the FA from an input file and another method to check if a sequence is accepted by the FA.

Parser

The main class of the LR(0) parser. It contains the grammar, the LR(0) parse table (which contains instances of TableEntry). When the parser is instantiated, the LR(0) table is automatically built. Other important methods of this class are:

- Closure
- Goto

- CanonicalCollection
- Parse
- CheckConflicts

ParserOutput

Class that constructs the parsing tree, by taking into consideration the father and sibling relation. It contains the root and the nodes of the tree. The most important methods are:

- createTreeFromSequence
- addChild
- addSibling
- printTree

Symbol Table

A class that uses a hash table with linked lists at each key. Unique for identifiers and constants (create one instance of ST)

It uses sum of ASCII codes of characters a a hashing function

- hash

- add

- remove

Scanner

The scanner is used in order to identify and separate the keywords, separators and operators from the program. It has lists for each of them, a PIF and SymbolTable (ST)

- readTokens

- tokenizeSpacelessChunk

- tokenizeLine

- runScanner

TableEntry

Object used to represent an entry in the table from the Parser class represented as a list. It has attributes like stateIndex, action as a string, reduceNonTerminal as string, reduceRHS as string and shifts as list

- reduceProductionString

TreeNode

Used to represent a node in the parsing tree, it has: index (string), info, parent, leftSibling, leftChild and level