FSM of the Lexical Analyzer (Higher Overview)

Chris Nutter
Jared Dyreson

September 16, 2020

Contents

1 Overview

This document explains the inner workings of the Lexical Analyzer from the perspective of the driver code from ‘lexi.cpp‘. Here, all components designed to make ‘lexi‘ a stand alone program work are collated into a single source document. The structure of this program is as follows:

1. Source file: where in memory is the file located and then extract the contents of it.
2. Rules: what set of parameters must the lexer abide by to produce the desired tokens.
3. Lexer: the actual mechanism that takes the content of the source document and applies rules to each line, producing tokens if matches are found.

2 Components of this FSM

The formal definition of an FSM includes the following:

1. Finite set of input symbols Σ: $\{A, B, C\}$ corresponding to the bullet points above.
2. Finite set of states Q: $\{D, G\}$
   (a) D: Unaccepted state. There were no matches and/or lexing errors were thrown.
   (b) G: Accepting state. There was at least one match to the rules presented and no lexing errors thrown.
3. Finite set of accepting states: $F \subseteq Q$
   (a) $F \subseteq \{G\}$
4. State-transition function(s) $N : (Q \times \Sigma) \to Q$
   (a) Each constructor and function call will advance the overall mechanism of Lexi

3 Graphical Representation

Each of the nodes correspond to the defined sets of states and input symbols.

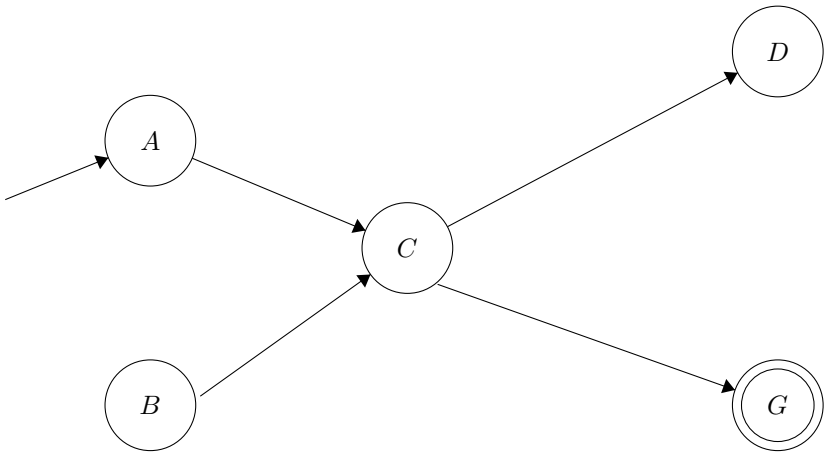

Figure 1: Lexi FSM