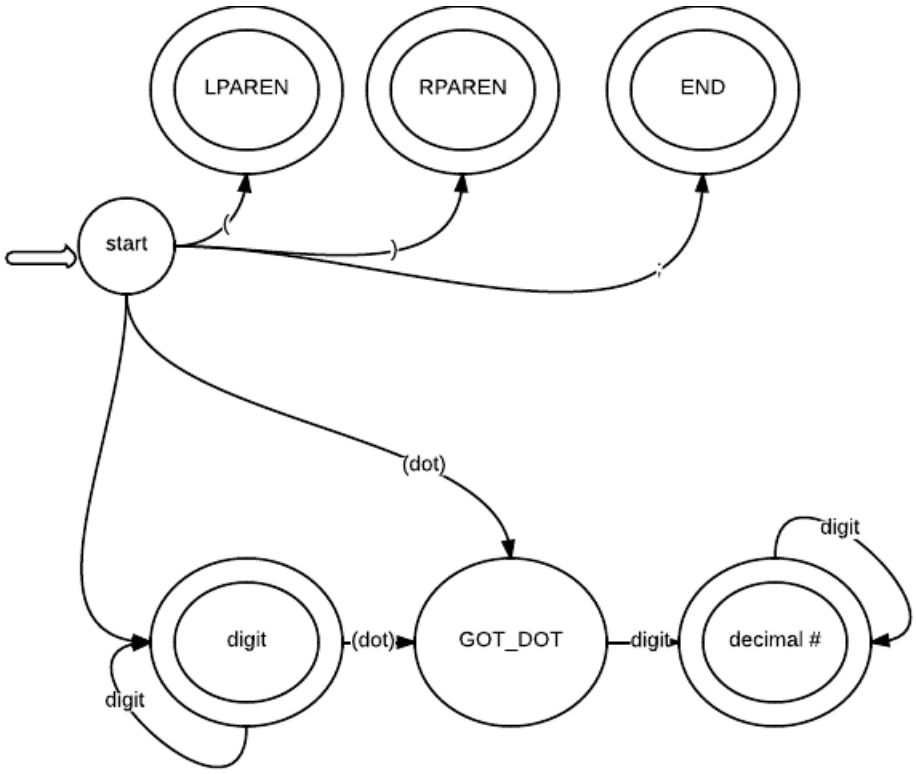
Dan Scarafoni

Write-up project 3 and 4

This project uses an existing scanner and parser code to build a calculator for simple mathematical functions. It read a file of expressions in the directory and evaluated them. It then placed the answers to the expressions in a separate file

Reader- this program reads in the text file and creates a linked list in which each node is a line from the file.

Scanner- The scanner goes through each character the strings stored at each node in the linked list produced by Reader. It uses this to create tokens of the character strings. The following deterministic finite automaton (DFA) was used for tokenization.



This machine was implemented using a switch statement, and as such has a big O(n) for run time.

Parser- the parser is the main “meat” meat of the program, and is in control of the scanner. The following grammar was used for the parser

StmntList → stmnt StmntList

stmnt → E;

E → T Etail

Etail → + T Etail

→ - T Etail

→ epsilon

T → F Ttail

F → (E)

→ digit

Ttail → * F Ttail

→ / F Ttail

→ epsilon

The FIRST and FOLLOW set of this grammar is

Production	FIRST	FOLLOW
StmntList	{(,digit}	{epsilon}
stmnt	{(,digit)}	{(,digit}
E	{(,digit}	{;}
Etail	{+,-}	{+,-}
T	{(,digit}	{+,-}
F	{(,digit}	{*,/}

Ttail {*,/} {+,-}

This grammar is, accordingly, LL(1), and as such has a complexity of bigO(n). Calculating the exact value of the expression is easy. An in-parser evaluator was added as an attempt for extra credit. This program evaluates the mathematical equations as they are being parsed.