Potential Fixedpoint Semantics for PEGREG

Jacob Salzberg

March 6, 2022

Progress on pegreg stalled because of concerns about the correctness of the algorithm that created an FST for possessive star.

Nonetheless, I believe it is still possible to define possessive star in finite state machines:

Parsing expression grammars are usually defined with respect to their input string, in effect embedding a "continuation" string into the semantics of a parsing expression grammar.

In the paper "Towards Typed Semantics for Parsing Expression Grammars" (2019), Rebeiro et. al. present an operational semantics for PEG that includes a left-recursive version of the star operator. I believe this semantics can be encoded as a fixpoint, viewing the strings "to be matched" as the set of all strings, then shown equivalent to the following rules:

$R_p[\![\langle \mathrm{ch}, \epsilon\rangle]\!] = \{\mathrm{ch}\}$	character literal
$R_p[\![\langle \mathrm{ch}, e_2\rangle]\!] = \{\mathrm{ch}.s \mid s \in R_p[\![\langle e_2, \epsilon\rangle]\!]\}$	character literal with continuation
$R_p[\![\langle e_1.e_2, e_3\rangle]\!] = R_p[\![\langle e_1, e_2.e_3\rangle]\!]$	concatenation
$R_p[\![\langle e_1/e_2, e_3\rangle]\!] = \{s_1 s_3, s_2 s_3 \mid s_1 \in R_p[\![\langle e_1, \epsilon\rangle]\!], s_3 \in R_p[\![\langle e_3, \epsilon\rangle]\!], s_2 \in R_p[\![\langle e_2, \epsilon\rangle]\!] \setminus R_p[\![\langle e_1, \epsilon\rangle]\!]\}$	ordered choice
$F_p[\![\langle e_1*, \epsilon\rangle]\!](X) = \{s_1 \mid s_1 \in R_p[\![\langle e_1, \epsilon\rangle]\!]\} \cup \{s_1 s_2 \mid s_1 \in X \wedge s_2 \in X\}$	greedy repitition
$R_p[\![\langle e_1*, \epsilon\rangle]\!] = \mathrm{lfp}\ F_p$	$*\epsilon$-case
$R_p[\![\langle e_1*, e_2\rangle]\!] = \{s_1 s_2 \mid s_1 \in R_p[\![\langle e_1*, \epsilon\rangle]\!], s_2 \in R_p[\![e_2]\!] \setminus R_p[\![e_1]\!]\}$	$*$general case

The set minus operation can then be encoded in a finite state machine: $L_1 \setminus L_2 = L_1 \cap \neg(L_2) = \neg(L_1 \cup \neg(L_2))$.