Generalized FSA, Context Free Grammar

Dhruva Sambrani

February 24, 2020

cont.

Let's black box a set of states to a single path.

Define the complexity of the FSA as the # of states. If we keep reducing the states, we'll eventually get to the simplest form.

Only one start state and only one accept state. This can always be done. If multiple starts, just put another state and take epsilons to the starts. Similarly from the finals to a single Final state.

Lets attempt to make it simpler (as defined above).

Defn: A genralised path b/w states is one that accepts a Regex instead of just a character.

Hence we are building a "Generalised" FSA, where δ is not over Σ, but rather over $\mathcal{P}(\Sigma)$. We also define a $\emptyset$ path(empty connection), which takes from q1 to q2 when it reads nothing.

Suppose fig 2402.1

Then, to remove Rk, change the path from qi to qj via Rij $\cup$ Ri $\circ$ Rk*$\circ$Rj

Now suppose Ri was $\emptyset$, then R'ij will also be $\emptyset$, so its a nice theoretic help.

Hence we can keep repeating to finally get a single regex.

Context Free Grammars

Consider the grammer of Arithmetic over +,-,*,/

Every expression can be broken into smaller expressions. The shortest expressions are those which are numbers.

Hence

$E \rightarrow E+E \mid E\text{-}E \mid E*E \mid E/E \mid (E) \mid N$

where,

$N \rightarrow 0 \mid 1 \mid ... \mid 9 \mid 0N \mid 1N \mid ... \mid 9N$

Hence checking if a string is E is basically a recursion to check if every part can be reduced to an expression.

(5+7) * 8

1. $E \rightarrow E * N$
2. $E \rightarrow (E) * N$
3. $E \rightarrow (N + N) * N$

0N1N

$E \rightarrow 01 \mid 0E1$

Defn

We can now define a context free gramm