Hybrid Syntax For Composition of Regular Languages

Christian Bendix Fjordstrøm

Motivation

Example

a*b|a

Solution

Hybrid syntax

Input split in grammar part and regex part

Convert grammar part to NFA templates

Preprocessing

Stratification

Creating NFA templates

Stratified grammar

NTs mutually recursive

depend only on NTs in previous layers

Tail position

Set of NTs where

- tail of NT is itself
- tail of (r1|r2) is union of tail of r1 and tail of r2
- tail of (r1 r2) where r2 is not ε is tail of r2
- tail of (r1 r2) where r2 is ε is tail of r1
- tail of other expressions is empty set

Restrictions for grammar to be stratified

mutual recursion only in the tail position

no mutual recursion in

- (r1&r2)
- (!r)
- (r*)

Stratification

Algorithm? - maybe too detailed?

Creating NFA templates

Explanation or example or both?

Conversion

Regex to NFA

NFA fragments from sub-expressions

Pass end-state as argument

Complement

Convert to DFA

Make DFA total

Flip accepting/rejecting

Intersection

Product construction

NFA to DFA

Subset construction

Minimisation of DFAs

Dead states

Split in groups

Check if consistent

Split into maximal consistent subgroups

Automata to Regex

State elimination

Matrix representation

Automata to Regex

Show example

Implementation

Testing

Property-based testing

Unit testing

Conclusion

Solution

Possible future work