Continuation of

Dhruva Sambrani

February 17-19, 2020

L∘L and L*

Building an NFA for these are explained in the book

L* cannot be proven to be regular by assuming it to be $\epsilon \cup L \cup L2 \cup ...$ because this is an infinite union, as infinite unions are not necessarily regular.

Building Regular Languages

Can we build a regular language with the following - "ϵ", singletons, {} and operations $\cup$, $\circ$, *? We know all languages built by such a construction are regular.

Thm: All regular languages can be built in such a way

Regular Expressions

Defn: R is a regex over Σ if R is

- a for some $a \in \Sigma$
- ϵ
- $\emptyset$
- finite R1$\cup$R2
- finite R1$\circ$R2
- R

Examples

If $\Sigma = \{0,1\}$ 1. 0 2. 1 3. 01 } O$\cup$1 4. (01) 5. 011$\circ$(0$\cup$1) 6. (0$\cup$1)$\circ$111 7. 011$\circ$(0$\cup$1)$\circ$111

Why regex?

Regex is a terse way to describe a DFA. A computer can then simulate the DFA and then do a pattern matching. Eg commands on linux-

- grep
- sed

Hence we try to build a regex from a DFA. Take a DFA, and break it down to the minimal states as above. We remove one state and replace it with a "black" box denoted by a set of regex's for all the possible state changes.

- forks can be a set of tuples (1a, 1b, 2a, 2b)
- loops can be replaced by a *.