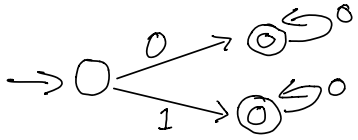
Lecture 4 summary

Main topic of this lecture is **Regular Expression** (Sometimes abbreviated as "**regex**")

Regex is shorthand for describing regular languages

> There are three different ways to describe regular languages, so far we have seen two of them, DFA, and NFA. And regex is the third.
>
> Ex. of regex:

$(0 \cup 1)\,0^*$ (This means a DFA that recognizes 0 or 1 followed by N number of 0's. Note, in this case N could be equal to or greater than 0.)



Def: A regular expression R is:

1.) a for some $a \in \Sigma$
2.) ε
3.) $\emptyset$
4.) $(R_1 \cup R_2)$ where R_1 and R_2 are regular expressions.
5.) $(R_1 \circ R_2)$ R_1, R_2 regex
6.) (R_1^*)

Shorthand

$R^+ \equiv RR^*$
$R^* \equiv R^* \cup \varepsilon$
$R^k \equiv \underbrace{RRR\ldots R}_{k \text{ times}}$

0^*1^* → This is describing any language that contains exactly one 1, and N number of 0's, where $N \geq 0$

$\Sigma^*1\Sigma^*$ → This is describing any language that contains at least one 1.

$\Sigma \equiv \{a \cup b \cup c \ldots \cup z\}$
Chars in Σ

$(\Sigma\Sigma)^*$ → This is describing languages of even length, including the empty string.

There are three operators in regular expression:

star = $*$
concatenation = $\circ$
union = $\cup$

Operator Precedence

star takes precedence over concatenation and union

concatenation takes precedence over union

union comes last

Normal precedence order can be changed with the use of parenthesis

The lecture ended after specifying how to prove that "if A is a regular language, then there is a regex for A"

The specification is given below.

Lemma: If A is regular then there is a regex for A.

Pf: Start with a DFA M for A

First construct a GNFA (Generalized NFA) for A

Modify the GNFA one state at a time to create the regex.