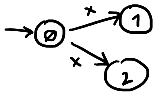
Transition Diagrams II

Tuesday, March 1, 2022 4:24 PM

FINITE NON-DETERMINISTIC AUTOMATA

↳ to vary the rules and not do them so strict

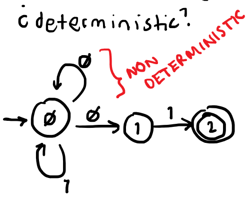


ND: when you have 2+ transitions with the same symbol

i.e.: $L = \{w \mid w \text{ finishes in "01"}\}$

$\Sigma = \{0, 1\}$

¿deterministic?



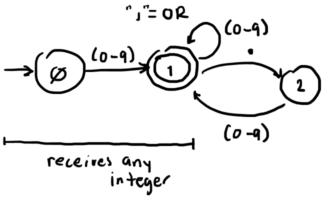
receives all strings ending with 01

In state 0, we receive a 0, where do we go? State 0 or state 1? AFND

$\delta(S_0, 0) = \begin{cases} S_0 \\ S_1 \end{cases}$? } IN ND, δ is not a function

i.e. AFND

$\Sigma = \{ \underbrace{(0-9),}_{\text{"," = OR}} . \}$ L = { numbers → integers, floats }



AFND $\xrightarrow{1:1}$ AFD (1 to 1 relationship)

All AFD are AFND

→ GRAMMARS

formal definition:

$G = \{V, T, s, R\}$

V → set of non terminal symbols

T → terminals

s → initial/start symbol

R → rules

i.e. What strings does it generate?

$\{x, y, xy, yx, xyx, yxy, \ldots\}$

All strings with x and y

→ A grammar generates a language.

↳ can we convert a grammar into an automata? each grammar has an automata

1. A state per non terminal
2. A connection per rule $A \to bc$: from state A to state c using b
3. Convert to final states the rules $A \to \lambda$ } empty

i.e. $S \to xX$

$S \to yY$

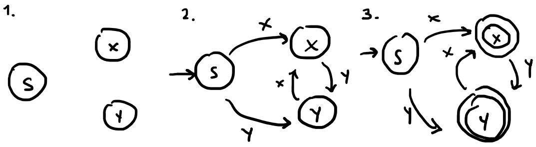
$S \to yY$

$X \to \lambda$

$Y \to xX$

$Y \to \lambda$

1. 2. 3.



Concatenation

$L_1 = \{x, yx\}$

$L_2 = \{yz, yy\}$

$L_1 \circ L_2 =$ strings that first have an element of L_1 followed by an element of L_2

$L_1 \circ L_2 = \{xyz, xyy, yxyz, yxyy\}$

Kleene Star

NOTATION
- ↳ union a and b : $a+b$
- ↳ concat a and b: $a \cdot b$
- ↳ Kleen of a : a^*

$L^* = L^0 \cup L^1 \cup L^2 \cup L^3 \ldots \infty$

$L^n = L \cdot L \cdot L \cdot L \ldots L$ } n times

$L = \{xy, yy\}$

↳ $L^0 = \{\}$

$L^1 = \{xy, yy\}$

$L^2 = \{xyxy, xyyy, yyxy, yyyy\}$

$L^3 = \{xyxyxy, \ldots\}$

$L^0 \cup L^1 \cup L^2 \cup L^3 \cup \ldots = L^*$

L^* = concat n times : regular (all automatas)

NOTATION:

AUTOMATA
- $0 \cdot \{\emptyset\}$
- $1 : \{1\}$
- $0+1 : \{\emptyset, 1\}$
- $(0+1)\cdot 0 : \{0,1\} \cdot \{0\}$
- $(0+1)\cdot 0 : \{00, 10\}$

i.e. 1) $(0+1)\cdot 0$

2) $\lambda + 1 = \{\lambda, 1\}$

$0 \cdot 1 = \{01\}$

$(0 \cdot 1)^* = \{\lambda, 01, 0101, 010101, \ldots\}$

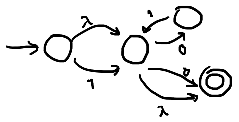
empty

$1 \cdot \lambda = \{1\}$

$1 \cdot \lambda = 1 = \{1\}$

$\lambda \cdot 1 = 1 = \{1\}$

3) $(\lambda+1) \cdot (01)^* \cdot (\lambda + 0)$



Context Independent Grammars

AFD ⟷ AFND
↕ ↕
ER ⟷ GR
} regular languages

stack automata ⟶ context independent grammar

→ Rules of form:

X → any combination of terminals and non-terminals

XX → Z (XX: context)   X → ~ (just one non terminal)

i.e.
- S → ZMNZ
- M → aMa
- M → Z
- N → bNb
- N → Z

What language do they generate?

zazabzbz

S → ZMNZ
→ ZaMaNZ
→ ZazaNZ
→ zazabNbZ
→ zazabzbz

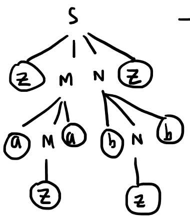
left derivation

S → ZMNZ
→ ZN bNbZ
→ ZMbZbZ
→ ZaMabzbz
→ Zazabzbz

right derivation

Same outcome

$\rightarrow ZaMabzbz$
$\rightarrow zazabzbz$



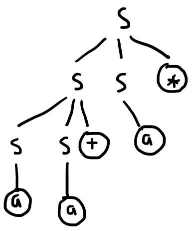
→ string derivation (it doesn't matter on which line it goes first - left or right)

It is the same derivation tree
↳ You can decide if left or right derivation
↳ We can decide to always use left derivation

i.e. Consider the Context Free Grammar (CFG):

$$S \rightarrow SS+ \mid SS* \mid a$$

build a derivation tree for the string aa+a*



we choose this child (production) so that the terminal (always one in the start symbol child) coincides with aa+a*

What language does this grammar generates?

$$L = \{ (a^{+} (+,*)^{*})^{*} \}$$