Math 322

(Q's) (15.1)

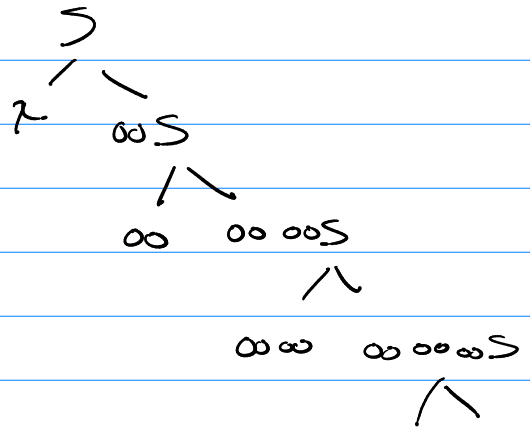
start symbol

$T = \{0, 1\}$ $N = \{S, A, B, \ldots\}$

a) $L(G) = \{$ all strings with only 0's and there is always an even number & then $\}$

goal: $L(G) = \{ \lambda, 00, 0000, 000000, \ldots \}$

$\{ S \to \lambda$
$S \to 00S \}$



$\{ S \to \lambda$
$S \to 0A0$
$A \to \lambda$
$A \to 0A0 \}$

(Q) Switch 01 to 10

~~01 → 10~~ not phrase-structure grammar

(Idea)

$S \to AS$
$S \to BS$
$\vdots$
$A \to 0$
$B \to 1$
$AB \to BA$

$S \Rightarrow AS \Rightarrow ABS$
$\Rightarrow$
$\Rightarrow AAA \ldots BBB$

#11 $T = \{0,1,2\}$ $N = \{S, A, B, C\}$

$P = \{$ $S \to 0SAB$, $S \to \lambda$, $BA \to AB$,

$0A \to 01$, $1A \to 11$, $1B \to 12$, $2B \to 22$ $\}$

$0^n 1^n 2^n$ $L(G) = \{\lambda, 012, 001122, 000111222, \ldots\}$

#11 use above to make 001122

$S \Rightarrow C \Rightarrow 0CAB \Rightarrow 00CABAB$

$\Rightarrow 00CAABB$

$\Rightarrow 00AABB$

$\overset{*}{\Rightarrow} 001122$

---

types of Grammars

type 0, phrase-structure, no restrictions (outside of left has some non-terminals)

type 1, context sensitive, $S \to \lambda$ is ok

all others do not contract

type 2, context free, (all the above)

and left side is a single non-term

type 3, regular, (all the above)

and right = a

or right = a A

#12 $S \to aAB$

$A \to Bb$

$B \to \lambda$ not ok

type 0 not type 1

$S \to aAB$ not 3
$A \to Bb$ not 3
$B \to bB$

type 0, type 1, type 2 not 3

$L(G)$ is the set of all terms such that $S \overset{*}{\Rightarrow} w$

$w \in T^*$

it is a context free language

Q's is $(x+y)-x$ a valid expression? yes

is $*x+(x-y)$ a valid expression? No

terminals are x, y

Non terms: variable, expression

V E ← start symbol

$P = \{ E \to (E),\ E \to E+E,\ E \to E*E$

$E \to E-E,\ E \to E \div E$

$E \to -E,$

$E \to V,\ V \to x,\ V \to y \}$

$(x+y)-x$

$E \Rightarrow E-E \overset{*}{\Rightarrow} (E)-V \overset{*}{\Rightarrow} (E+E)-x$

$\overset{*}{\Rightarrow} (x+V)-x$

$\overset{*}{\Rightarrow} (x+y)-x$

Backus-Naur Form

non-term <name>

term name

left → right left := right

and

left → r_1 left := r_1 | r_2 | r_3

or left → r_2 or or

or left → r_3

More example

<variable> := x | y

<expression> := (<expression>) | <expression> + <expression> | <variable> | ...etc

---

13.2 Finite-State Machines with output

(graphs that handle input, output, recognizing strings)

→ "knowledge" of the graph

(ex) 2¢ candy machine

Idea: graph = vertices (+) edges

"knowledge" of machine

moving from known to known events based on input / output

changes

+1

old vertex

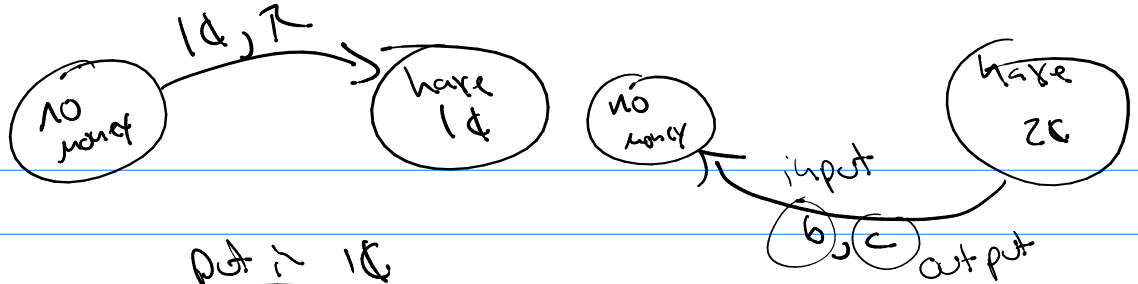
or

known state #1

new vertex

know state #1

#2



put in 1¢

transaction

hit candy button: b

get candy: c

output

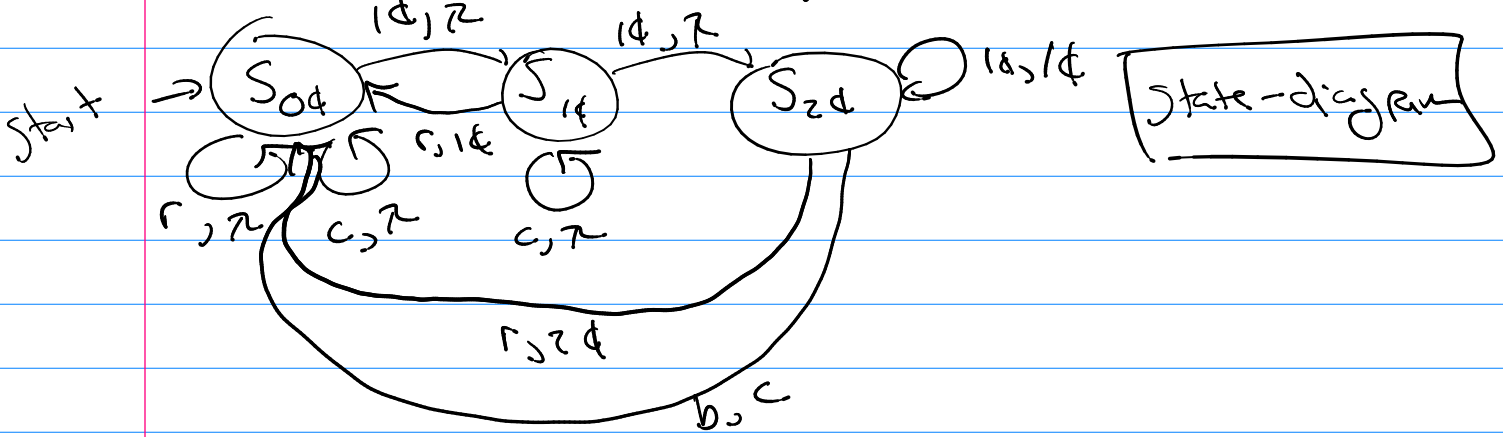
---

all together

I: 1¢, r, b

↑ return $ ↑ get candy

O: r, 1¢, 2¢, c



state-diagram

State-Table

f: transition function g: output function

States	I = i_1 i_2 i_3 .. i_n	i_1 i_2 .. i_n = I
S_0		
S_1	newstate	
S_2		output for (S_2, i_1)
:		
S_k		

input

---

ex

2¢

	f: 1¢, r, b	g: 1¢, r, b
$S_{0¢}$	$S_{1¢}$ $S_{0¢}$ $S_{0¢}$	r r r
$S_{1¢}$	$S_{2¢}$ $S_{0¢}$ $S_{1¢}$	r 1¢ r
$S_{2¢}$	$S_{2¢}$ $S_{0¢}$ $S_{0¢}$	1¢ 2¢ c

$$M = (S, I, O, f, g, s_0)$$

Finite State Machine with output.

S : finite set of states

$s_0 \in S$: start state

I : inputs

O : outputs

$f : S \times I \to S$ transition function

$g : S \times I \to O$ output function

What can these do?

(1) I/O machine (candy, pay pads, ...)

(2) Delay

ex) 00111001....

delay by 00

00101110 01...