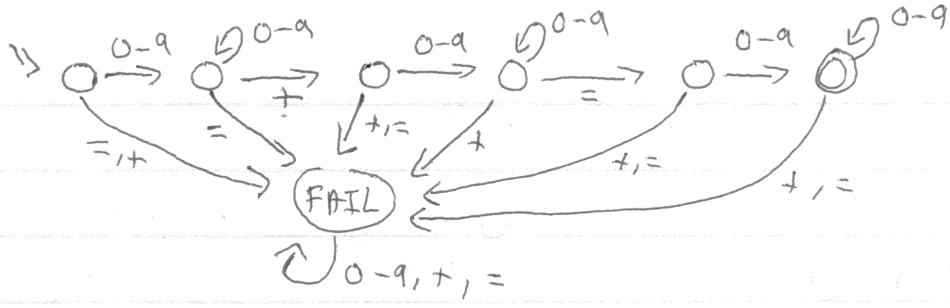
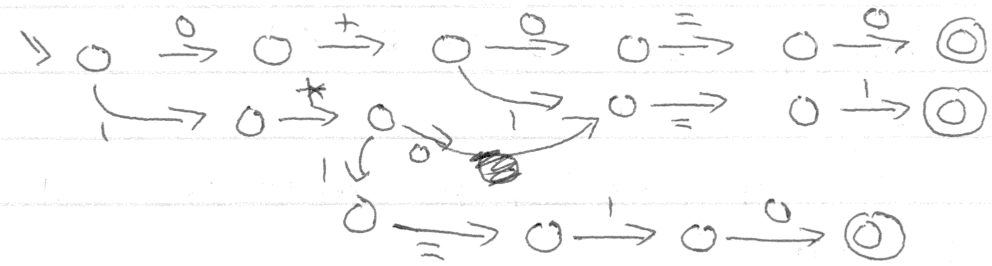
2-4/

Well-formed addition equations:

$\Sigma = 0\text{-}9, +, =$

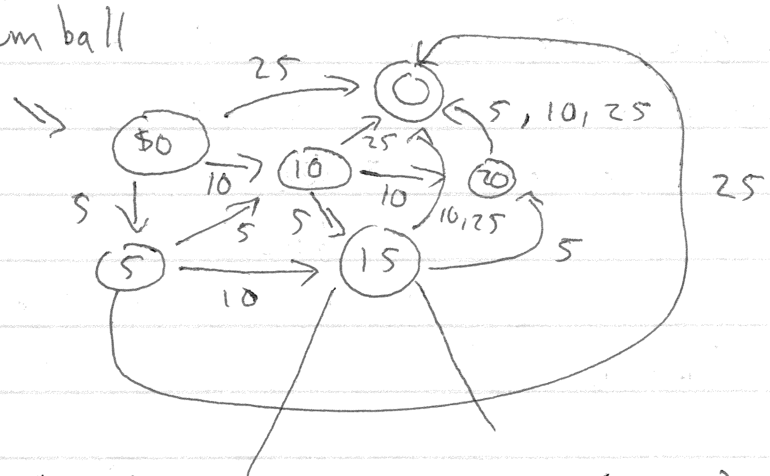


Correct, 1-bit additions



{0+0=0, 0+1=1, 1+0=1, 1+1=10}

Gumball



(001) 5 ↦ 20 (0 1)

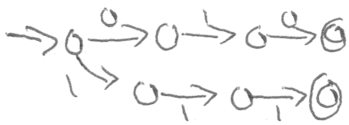
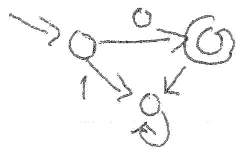
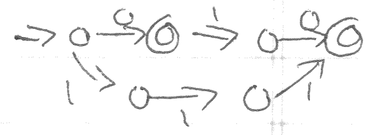
(010) 10 ↦ 25 (1 0)

(100) 25 ↦ 25 (1 0)

3-1/ Language := Set (Strings)

union of two languages?

$\{010, 111\} \cup \{0\} = \{0, 010, 111\}$

$\forall A, B \in REG.\ \exists C \in REG.\ C = A \cup B$?

closure property (of $\cup$ in REG)

what does $X \in REG$ mean?

REG = regular languages = languages accepted by a DFA

$\exists d \in DFA.\ L(d) = X$

$A \Rightarrow \langle \Sigma, Q_A, q_{0A}, \delta_A, F_A \rangle$ Given

$B \Rightarrow \langle \Sigma, Q_B, q_{0B}, \delta_B, F_B \rangle$

make $\langle \Sigma, Q_C, q_{0C}, \delta_C, F_C \rangle \Rightarrow C$

a string x should be accepted if A accepts or B accepts

is $Q_A \times Q_B$ a finite set? $= Q_C$

$(q_{0A}, q_{0B}) = q_{0C}$

~~$F_A \times F_B$~~ $= F_C$ (intersection)

$F_A \times Q_B \cup Q_A \times F_B$

$$\delta_C((q_a, q_b), c \in \Sigma) = (q_a', q_b')$$

$$q_a' = \delta_A(q_a, c)$$

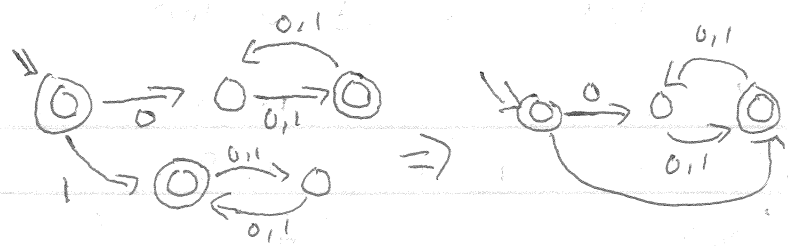
$$q_b' = \delta_B(q_b, c)$$

3-2

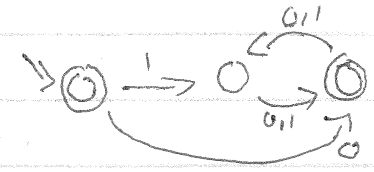
A = { starts with 0, even # or st 1, odd } or mt

B = { starts with 1, ~~odd~~ even # or st 0, odd } or mt

A =



B =



A = < Q_a = { X, Y, Z }

q_{0a} = X

F_a = {X, Z}

δ_A =	0	1
X	Y	Z
Y	—	Z
Z	—	Y

B = < Q_B = { L, M, N }

q_{0B} = L

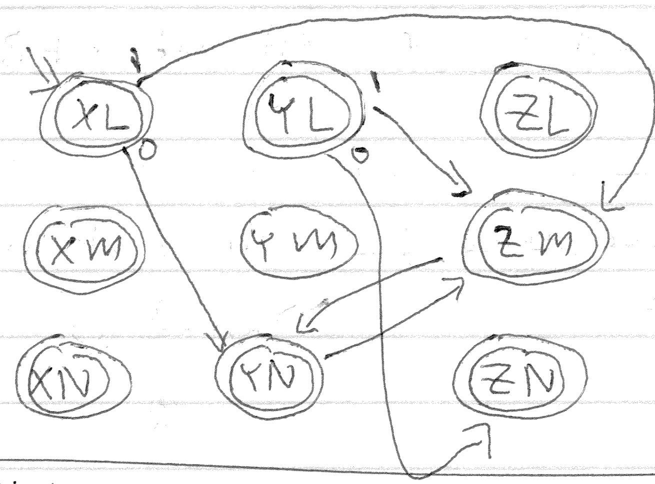
F_B = {N}

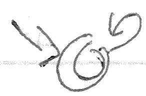
δ_B =	0	1
L	N	M
M	—	N
N	—	M

C = < Q_c = { XL, YL, ZL
XM, YM, ZM
XN, YN, ZN }

q_{0c} = XL

F_c = ZL, ZM, ZN,
XN, YN, ZN
XL XM XN
XL YL ZL





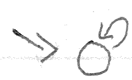
CLAIM: FIN ⊂ REG

If A ∈ FIN, A = { x_0, ..., x_n } or = { }

Break A into {x_0} and {x_1 ... x_n}

x_0 = c_0 ... c_n

IND



Union

3-3/ Regular Operations are set-operations that REG is closed under.

Union – proved $A \cup B$

Intersect – proved $A \cap B$

Complement $\bar{A}$ or A^c

$x \in \bar{A} \iff x \notin A$ q^* is the final state when running a machine

$\bar{A}$ <u>does</u> accept x A doesn't accept x

$q^* \in F_{\bar{A}}$ $q^* \notin F_A$

state of DFA is $\log_2 Q$ bits

$$\bar{A} = \langle \Sigma, Q_A, q_{0A}, \delta_A, Q_A - F_A \rangle$$

Star – $A^* = \varepsilon \cup A \circ A^*$

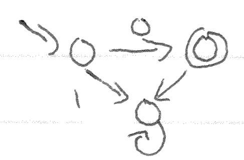
Concatenate – $A \circ B$ ($xy \in A \circ B$ iff $x \in A$ and $y \in B$)

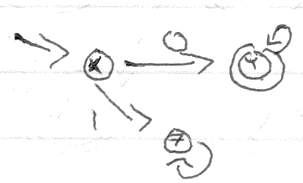
Reversal – A^R

Difference – $A - B$ ($x \in A - B$ iff $x \in A$ and $x \notin B$)

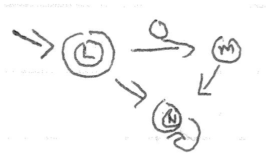
$= A \cap \bar{B}$

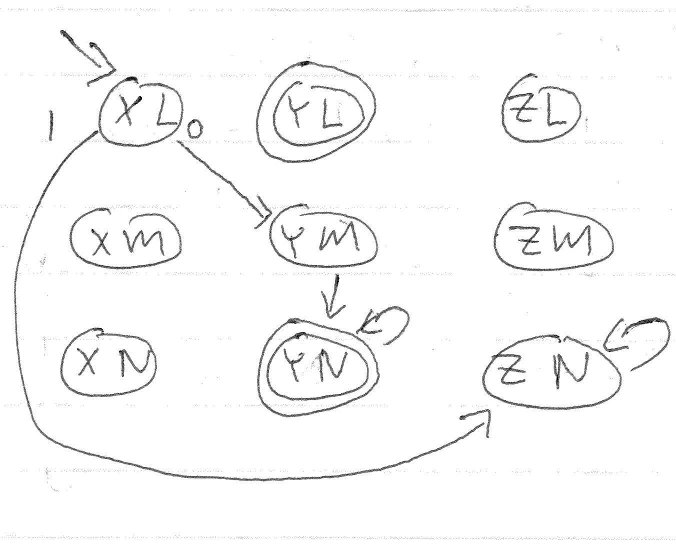
$B = \{ 0 \}$ ~~$A = \{$ everything $\}$~~ $A = \{$ starts with 0 $\}$





$\bar{B}$





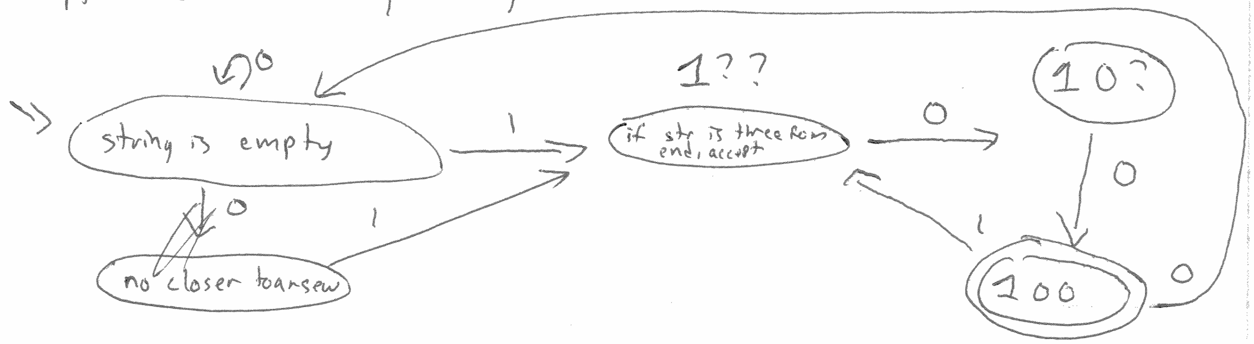
3-4/

```
typedef enum { XL, YM, YN, ZN } state_t;

int machine ( )   {

    state_t  st  = XL;
    while ( char c = getc() ) {
        switch (st) {
        case XL:  switch (c)  {
                    case '0' :  st = YM;   break;
                    case '1' :  st = ZN;   }
        case YM :  st = YN;  b;
        case YN :  st = YN;  b;
        case ZN :  st = ZN;  b;  } }
    return  st == YN;
```

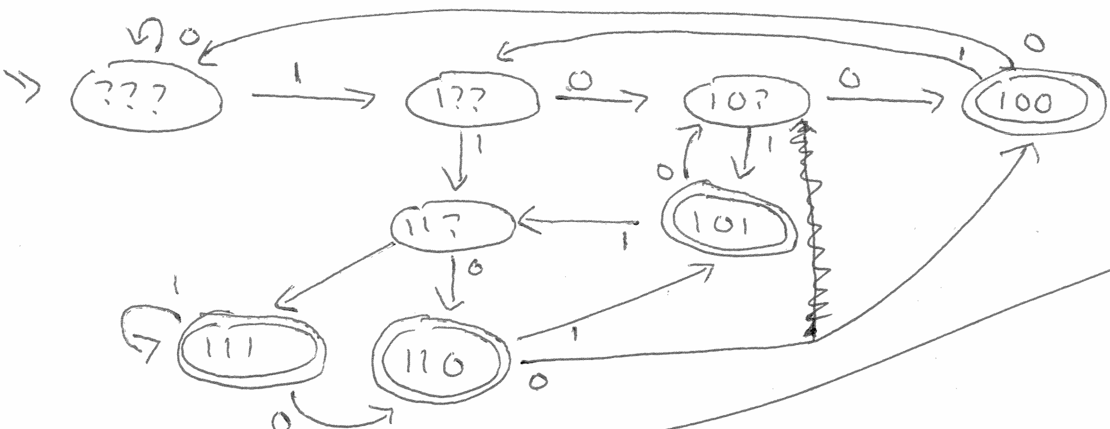
Regexp : $\underbrace{\text{courses}}_{x} / \underbrace{*}_{\text{any}^*} / \underbrace{\text{grades}}_{y} / \underbrace{*}_{\text{any}^*} , \underbrace{\text{fail}}_{z}$

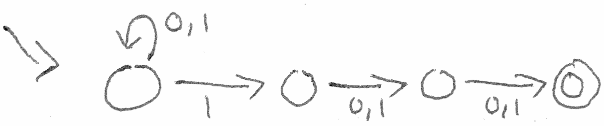
$$re = x \circ \Sigma^* \circ y \circ \Sigma^* \circ z$$

1-1/ A = { all binary strings where third-to-last character is 1 }



0100 ✓
1000 X
11100 ✓





NFA - Non-deterministic Finite Automata

diff 1 : states don't have all transitions

diff 2 : states may have >1

states can any number of trans. per character

Future telling
backtracking
fork()-ing
mystery good one

$a \xrightarrow{c} b$ $\quad \delta(a, b) = c$

DFA $d = \langle \Sigma, Q, q_0 \in Q, \delta : Q \times \Sigma \to Q, F \subset Q \rangle$

NFA $n = \langle \Sigma, Q, q_0 \in Q, \delta : Q \times \Sigma_\varepsilon \to P(Q), F \subset Q \rangle$

$a \xrightarrow{c} b$, $a \xrightarrow{c} d$ $\quad \delta(a, c) = \{b, d\}$

a $\quad \delta(a, c) = \{\}$

$\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$ epsilon

$a \xrightarrow{\varepsilon} b$ $\quad \delta(a, \varepsilon) = \{b\}$