To-do:

- ☐ HW1 - Due ~ 1 Wk before Quiz.
  - ↳ Sections 1.1, 1.2

Hw: 20% (4 ass) — no late ass

Q1: 10% - in class Jan 29

Midterm: 25% - Fri 18:30-20:00 Mar 1

Q2: 15% Mar 14 (in class)

Final: 30%

cs.uwaterloo.ca/~watrous/c53

John Watrous

QnC 3312

OH: Tues 1-3 pm or by appt

Theory of Computing

mathematics of abstract models of computation, and the problems they can (or cannot) solve.

Foundations & Defs:

- Assume comfortable with set theory
- finite Set S has size |S|
- Sets can be infinite

i.e: $\mathbb{N} = \{0, 1, 2, \ldots\}$ (natural numbers)

$\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ (integers)

$\mathbb{Q} = \{x : x = n/m, n, m \in \mathbb{Z}, m \neq 0\}$ (rational #s)

$\mathbb{R}, \mathbb{C}$ real { complex #s.

These are inf sets, but we need a more refined notion

Def: A set A is countable if there exists any one-to-one (injective) function f of the form:

$$f: A \to \mathbb{N}$$

These are equivalent (for Any set A)

1. A is countable
2. either A is empty, or there is an onto (or surjective) function

$$g: \mathbb{N} \to A$$

3. Either A is finite, or there exists a one-to one and onto (or bijective) function.

$$h: \mathbb{N} \to A$$

Ex.

$\mathbb{N}$ is countable

$\mathbb{Z}$ is countable

$f: \mathbb{Z} \to \mathbb{N}$ def as:

$f(0)=0$, $f(n)=2n$,

$f(n)=2n+1$

for all $n \geq 1$

$\mathbb{Q}$ is countable

0 ← list 0

$-1, +1$ ← list 1

$-2, \frac{-1}{2}, \frac{1}{2}, 2$ ← list 2

$-3, \frac{-3}{2}, \frac{-2}{3}, \frac{-1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{3}{2}, 3$ ← list 3

⋮

{ . . . . . } list k

sorted list of all fractions

$\pm \frac{n}{m}$ $1 \leq n, m \leq k$,

either $n=k$ or $m=k$

and $\gcd(n,n)=1$

Is every set countable? No.

For any set A, the power set of A is the set of all subsets of A.

Notation: $P(A)$ $A = \{1, 2, 3\}$

$P(A) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$

The set $P(\mathbb{N})$ is not countable

Pf:

- Assume it is countable (toward contradiction)
- This implies there exists an onto function

$$g: \mathbb{N} \to P(\mathbb{N})$$

Define $S = \{n \in \mathbb{N} : n \notin g(n)\}$

g is onto, and $S \in P(\mathbb{N})$ so there must exist at least one number $n \in \mathbb{N}$ so that $S = g(n)$.

Is $n \in S$?

Is $n \in S$?

$n \in S \Leftrightarrow n \in g(n)$ (because $S = g(n)$)

$n \in S \Leftrightarrow n \notin g(n)$ (definition of S)

So

$$n \in g(n) \Leftrightarrow n \notin g(n)$$

Contradiction.

Comments:

- Proof technique is called diagonalization

Terminology:

Alphabet: Any Finite, non-empty set. We view these as symbols

A string over an alphabet Σ is a finite, unordered sequence of symbols from Σ →

0110 – String

$0110\ldots$ – not a string

ε – empty string

length of a string is the total number of symbols.

$0110 \rightarrow$ length $= 5$

$\varepsilon \rightarrow$ length $= 0$

$|w|$ is the length of string w.

$|0110| = 5$

A language over an alphabet Σ is a set of strings over Σ

$L = \{w : w \text{ is a string over } \{0,1\}, |w| \text{ is even}\}$

For any alphabet Σ, the language containing all strings over Σ is countable

$\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, \ldots$

$0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad \ldots$

A set A is countable iff it is the empty set, or there is an onto function

$f: \mathbb{N} \to A$

Σ = alphabet

Σ^* = set of all strings over Σ

↳ For $\Sigma = \{0,1\}$ Σ^* is countable

$\varepsilon, 0, 1, 00, 01, 10, 11, \ldots$

Proposition: Σ alphabet
$\Rightarrow P(\Sigma^*)$ is not countable

Cantor: $P(\mathbb{N})$ is uncountable.

Have a bijection $f: \mathbb{N} \to \Sigma^*$
Define $g: P(\mathbb{N}) \to P(\Sigma^*)$

$$g(A) \stackrel{\text{def}}{=} \{f(n) : n \in A\}$$

Since $P(\mathbb{N})$ is uncountable, and there is a bijection $P(\mathbb{N}) \to P(\Sigma^*)$, then $P(\Sigma^*)$ is uncountable as well

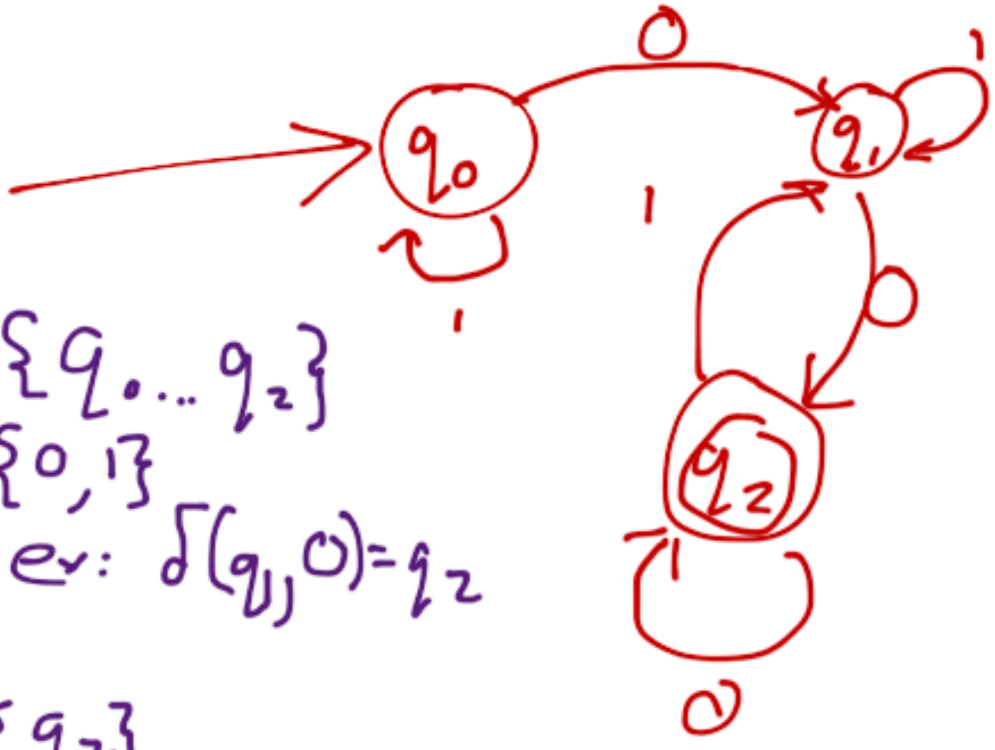
DFAs

A DFA is a 5-tuple

$$M = (Q, \Sigma, \delta, q_0, F)$$

where

- Q: a finite, non-empty set whose elements we call states
- Σ is an alphabet
- δ is a function (a transition function) of the form: $\delta: Q \cdot \Sigma \to Q$,
- q_0 is the start state.
- F, $F \subseteq Q$ is a subset of states - "accept states"



$Q = \{q_0 \dots q_2\}$

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

$\delta = \dots$ ex: $\delta(q_1, 0) = q_2$

$q_0 = \dots$

$F = \{q_2\}$

Definition:

Supp: $M = (Q, \Sigma, \delta, q_0, F)$ is a DFA and $w \in \Sigma^*$. The DFA M accepts w iff

1. $w = \varepsilon$, and $q_0 = F$, or
2. $w = \sigma_1 \ldots \sigma_n$ for $n \geq 1$, $\sigma_1 \ldots \sigma_n \in \Sigma$ and there exist states $r_0 \ldots r_n \in Q$ such that $r_0 = q_0$, $r_n \in F$ and $\delta(r_k, \sigma_{k+1}) = r_{k+1}$ for all $k \in \{0 \ldots n-1\}$

An equivalent def:

- extend δ to operation $r \in \Sigma^*$.

Define:

$\delta^*: Q \times \Sigma^* \to Q$

as follows:

1. $\delta^*(q, \varepsilon) = q$
2. $\delta^*(q, w\sigma) = \delta(\delta^*(q, w), \sigma)$ for all $q \in Q$, $w \in \Sigma^*$, $\sigma \in \Sigma$

M accepts w iff $\delta^*(q_0, w) \in F$

If M does not accept w, we say it rejects w.

The lang recognized by M is:

$$L(M) = \{w \in \Sigma^* : M \text{ accepts } w\}$$

Definition:
A language $A \subseteq \Sigma^*$ over any alphabet Σ is regular iff there exists a DFA M with $A = L(M)$

How Many regular languages are there (over a given Σ?) → Countable.

$n \cdot n \cdot 2^{n|\Sigma|}$

↑ f. exists ↑ accept

NFAs

A NFA is a 5-tuple

$N = (Q, \Sigma, \delta, q_0, F)$

where

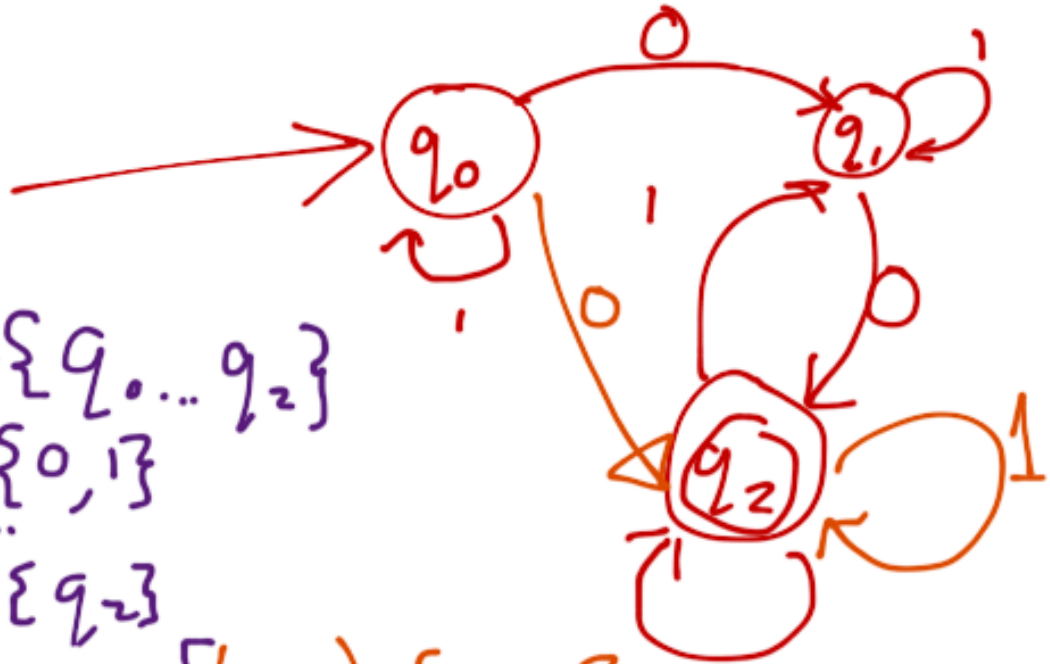
→ Q: a finite, non-empty set whose elements we call states

→ Σ is an alphabet

→ q_0 is the start state.

→ F, $F \subseteq Q$ is a subset of states - "accept states"

→ $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to P(Q)$



$Q = \{q_0 \dots q_2\}$

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

$q_0 = \dots$

$F = \{q_2\}$

$\delta = \dots$ ex: $\delta(q_0, 0) = \{q_1, q_3\}$?

Definition:

Supp: $N = (Q, \Sigma, \delta, q_0, F)$ is a NFA and $w \in \Sigma^*$. The NFA N accepts w iff there exist:

1. $m \in \mathbb{N}$
2. $r_0, \ldots r_m \in Q$, and
3. $\sigma_1, \sigma_2, \ldots \sigma_m \in \Sigma \cup \{\varepsilon\}$

such that

$r_0 = q_0$, $r_m \in F$, $w = \sigma_1 \ldots \sigma_m$, and

$$r_{k+1} \in \delta(r_k, \sigma_{k+1})$$

for all $k \in \{0, \ldots, m-1\}$

Alternatively, $\delta^*: Q \times \Sigma^* \to P(Q)$ as follows:

For any $R \subseteq Q$, define

$$E(R) = \{q \in Q : q \text{ can be reached from one } r \in R \text{ by following } \geq 0 \text{ } \varepsilon \text{ transitions}\}$$

$$\delta^*(q, \varepsilon) = E(\{q\})$$

$$\delta^*(q, w\sigma) = E\left(\bigcup_{r \in \delta^*(q,w)} \delta(r, \sigma)\right)$$

Now, N accepts w iff

$$\delta^*(q_0, w) \cap F \neq \emptyset$$

$$L(N) = \{w \in \Sigma^* : N \text{ accepts } w\}$$

Thm: Let Σ be an alphabet and let $A \subseteq \Sigma^*$ be a language.
A is regular iff $A = L(N)$ for NFA N.
i.e. NFA $\leftrightarrow$ DFA conversion

→ Given NFA $N = (Q, \Sigma, \delta, q_0, F)$
want DFA $M = (P, \Sigma, \eta, p_0, G)$
so that $L(N) = L(M)$

Step 1: construction
Step 2: prove it works.

Continuing...

Suppose $N = (Q, \Sigma, \delta, q_0, F)$ is an NFA. It follows that $L(N)$ is regular.

Want: DFA $M = (P, \Sigma, m, p_0, G)$ such that $L(m) = L(N)$

Iff it has an NFA or DFA, it is regular

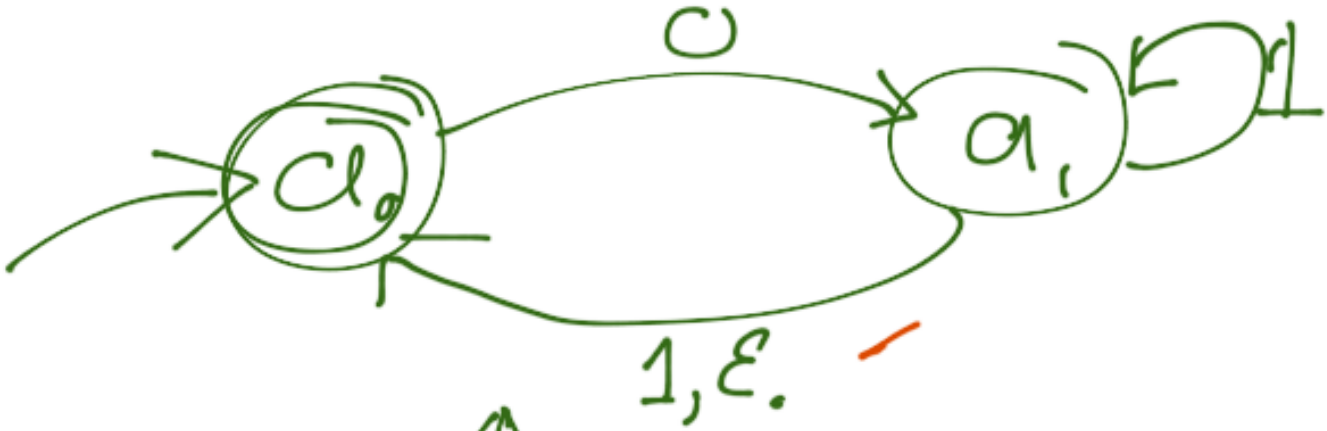
1. construct M.
2. prove $L(M) = L(N)$

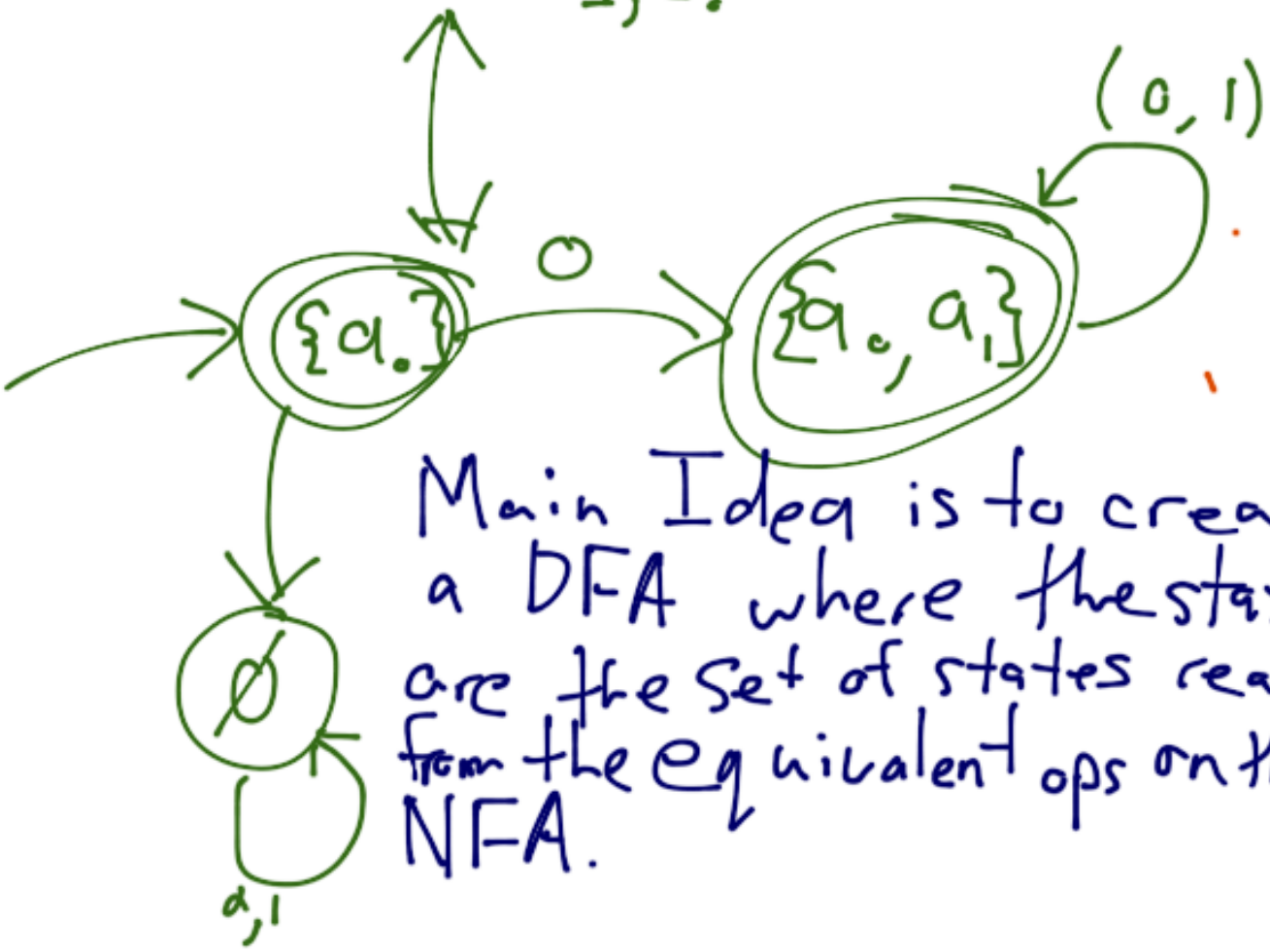
1. Construct M

- $P = \mathcal{P}(Q)$ (power set) $\{p \subseteq Q, E(P) = P\}$
- $p_0 = E(\{q_0\})$ (epsilon closure)
- $G = \{p \in P : p \cap F \neq \emptyset\}$

(all possible states that could have been after this sequence)

- $m(p, \sigma) = E\left[\bigcup_{q \in P} \delta(q, \sigma)\right]$





Main Idea is to create a DFA where the states are the set of states reachable from the equivalent ops on the NFA.

$\delta^*(E(p), w) = \bigcup_{q \in P} \delta^*(q, w)$

for all $p \in P, w \in \Sigma^*$ } if we prove this, we prove the DFAs are equivalent, i.e. the accept states are identical

$A = B$ iff

$A \subseteq B$ and $A \supseteq B$

Regular Operations: (on languages)

→ maintain regularity.

Alphabet Σ, $A, B \subseteq \Sigma^*$

1. Union:

$$A \cup B = \{w \in \Sigma^* : w \in A \text{ or } w \in B\}$$

2. Concatenation:

$$AB = \{w \in \Sigma^* : w = uv, \text{ for some choice of } u \in A \text{ and } v \in B\}$$

3. Kleene Star:

If $A \subseteq \Sigma^*$ then

$$A^* = \{\varepsilon\} \cup A \cup AA \cup \ldots = \bigcup_{n \in \mathbb{N}} A^n$$

The idea is that A^* is the set of all concatenations of A.

Thm: The regular Languages are closed under the regular operations.

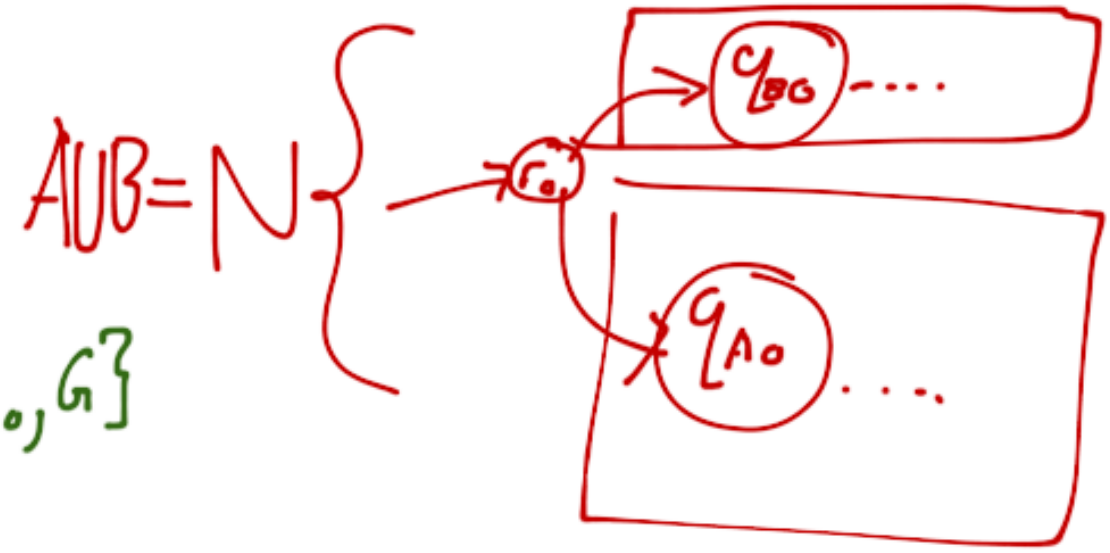
Σ alphabet, $A, B \subseteq \Sigma^*$ regular.

$\Rightarrow$ $A \cup B$ regular,
AB regular
A^* regular.

Pf: A, B regular
$\Rightarrow$ exists DFAs
$M_A = (Q_A, \Sigma, \delta_A, q_{A0}, F_A)$
$M_B = (Q_B, \Sigma, \delta_B, q_{B0}, F_B)$
such that:
$A = L(M_A)$, $B = L(M_B)$

No loss of generality in assuming
$Q_A \cap Q_B = \emptyset$

AUB:



$N=[R,\Sigma,m,r_0,G]$

$R=\{r_0\}\cup Q_A\cup Q_B$

$G=F_A\cup F_B$

$m(q_A,\sigma)=\bigcup_{q_A\in Q_A}\{\delta(q_A,\sigma)\}$] this is for m_A, m_b is similar.

$m(r_0,\varepsilon)=\{q_{A0},q_{B0}\}$

$m(r_0,\sigma)=\emptyset$

The idea is that we create m to be the transition fn in N to map to Both A and B

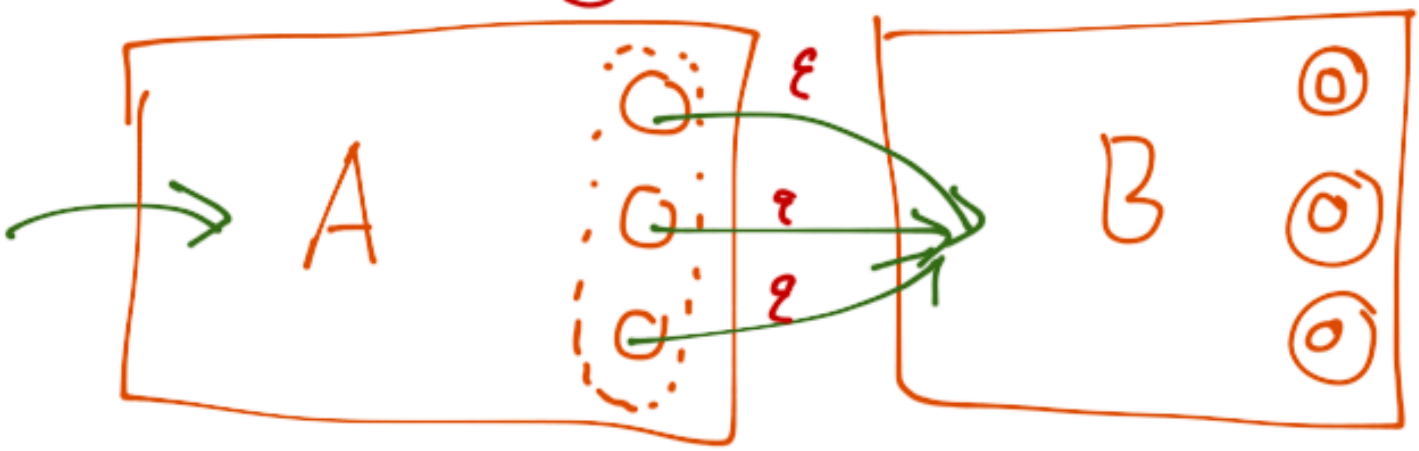
Alternatively, the cartesian Product Construction:

$$M = (Q_A \times Q_B, \Sigma, \mu, (q_{A0}, q_B), G)$$

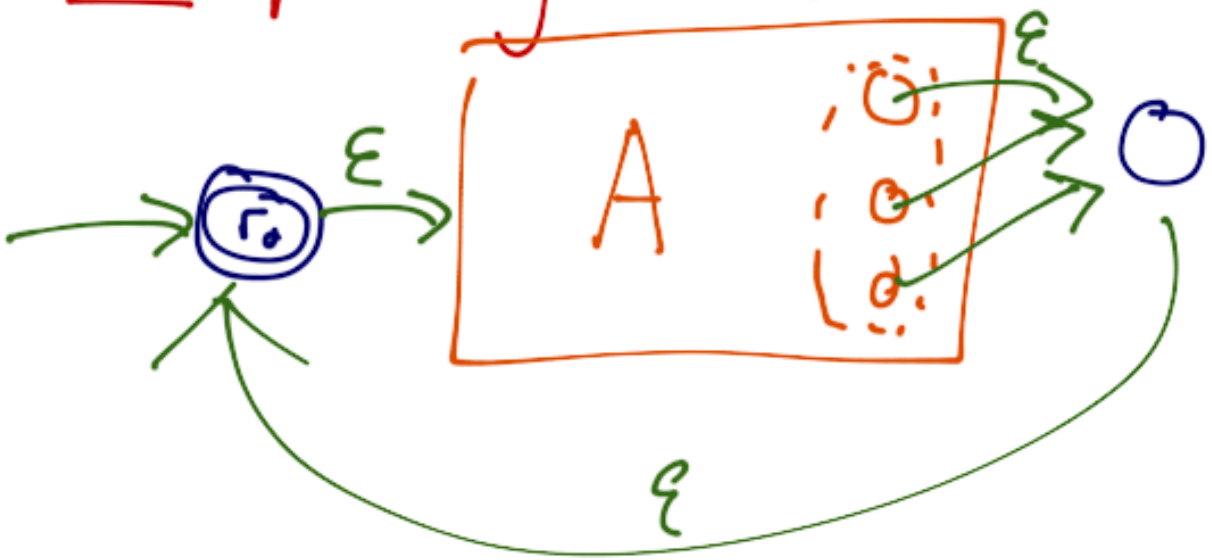
$$G = \{(q_A, q_B) : q_A \in F_A \text{ or } q_B \in F_B\}$$

$$M = ((q_A, q_B), \sigma) = (\delta(q_A, \sigma), \delta(q_B, \sigma))$$

AB: (proof by picture)



A^*: (proof by Picture)

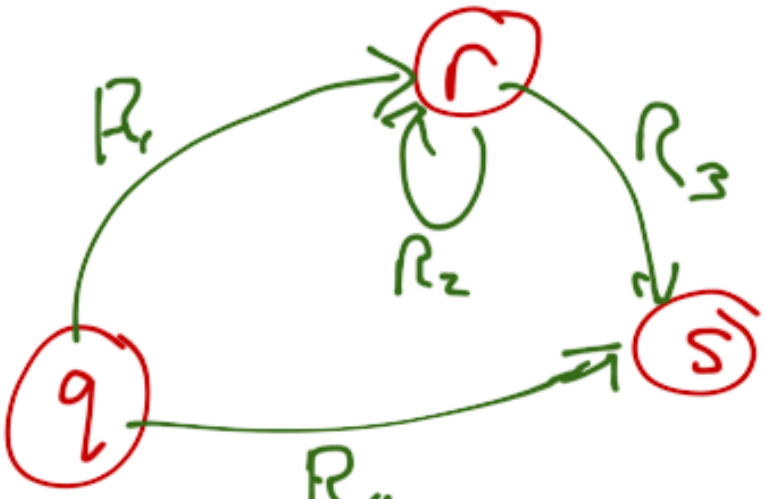


Assignment 1:

1. a) If we define $g: \mathbb{N} \to B$ as the same function as $f: \mathbb{N} \to A$ in the assumption, satisfying the requirement of the onto function. From this we can draw the conclusion that the subset of any countable set is also countable.

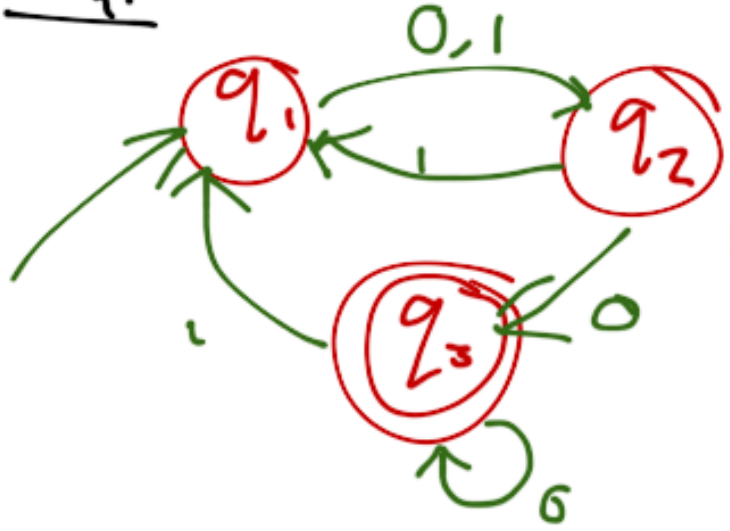
b)

Left Blank (Usec)

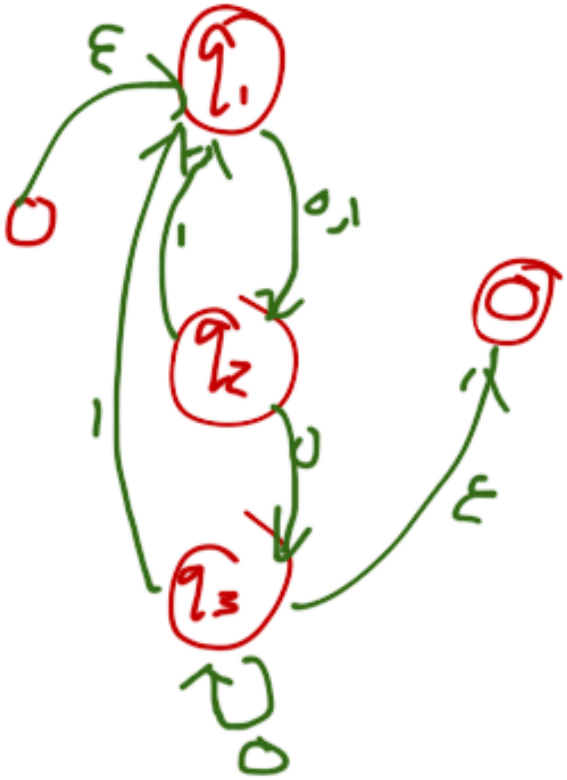


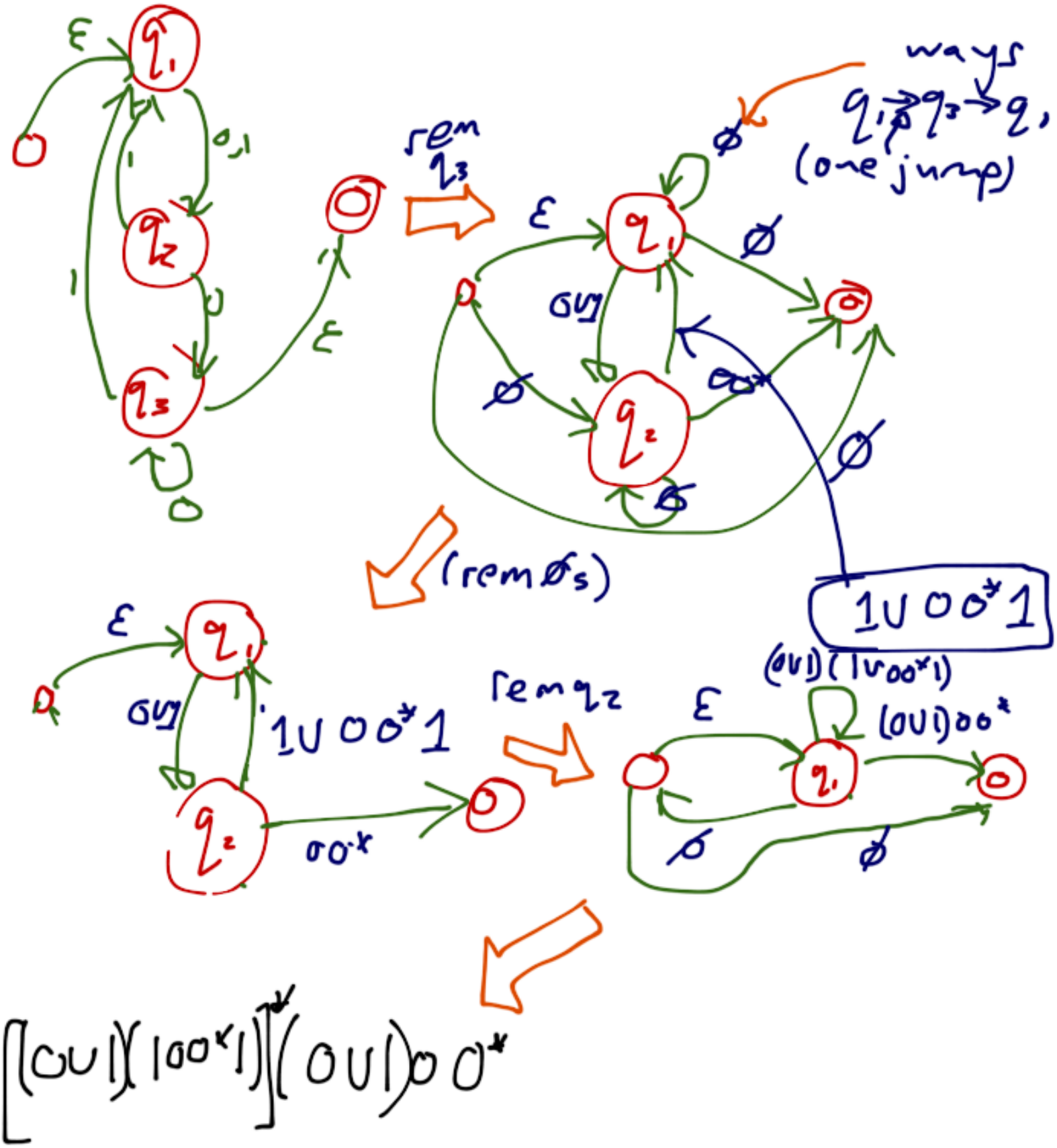
⇒ remove r

Ex:



⇒




ε
q1
0,1
1
q2
0
q3
0
1
ε
rem q3
ways q1 → q3 → q1 (one jump)
ε
q1
∅
0
0U1
∅
∅
q2
00*
∅
∅
(rem ∅s)
1U00*1
ε
q1
0U1
1
q2
00*
1U00*1
rem q2
(0U1)(1U00*1)
ε
(0U1)00*
q1
∅
∅
[(0U1)(100*1)]*(0U1)00*

Techniques For proving Certain Languages are not regular:

Pumping Lemma:

If you have a DFA with P (or fewer) states, and you read a string with at least P symbols, you must have visited the same state twice.

If you have a string where $len(str) \geq len(states)$ there exists a shorter string that arrives at the same state.

Supp. $A \subseteq \Sigma^*$ is regular. There exists an integer $p \geq 1$ (pumping length of A) that satisfies this property:

(# of states)

For every $w \in A$ with $|w| \geq p$ there exists strings $x, y, z \in \Sigma^*$ such that

1. $y \neq \varepsilon$
2. $|xy| \leq p$
3. $xy^i z \in A$, for all $i \in \mathbb{N}$.

Proof: (sketch)

Pumping Lemma (YT video)

If L is a regular language.

$\exists n \Rightarrow$ # states in machine for L

$\forall z \in L$, where $|z| \geq n$

$\exists v, w, x$ such that $z = vwx$, such that $|vw| \leq n$ and $|w| \geq 1$

Then: $\forall i \geq 0$, $vw^i x$ is also in L

↑

The idea is that we want to disprove a language from being regular, and can do that by coming up with a z that satisfies some values of i, then using those to disprove the language of being regular.

- regular → pumping
- ¬pumping → ¬regular

¬Pumping Property:

$\forall n$ where n = # states in language L

$\exists z \in L$, $|z| \geq n$

$\forall v, w, x$ where $z = vwx$

and $|vw| \leq n$

and $|w| \geq 1$

$\exists\, i \geq 0$ such that $vw^i x$ is <u>not</u> in L

$\Rightarrow L$ is not regular

Ex:

$A = \{0^n 1^n : n \in \mathbb{N}\}$

$= \{\varepsilon, 01, 0011, 000111, \ldots\}$

Claim: A is not regular.

Proof. Assume A is regular, toward contradiction. By the Pumping Lemma, there exists a pumping length p for A.

Let $w = 0^p 1^p$

We have $w \in A$ and $|w| = 2p \geq p$.

There must therefore exist x, y, z such that:

1. $y \neq \varepsilon$
2. $|xy| \leq p$
3. $xy^t z \in A$ for all $t \in \mathbb{N}$

$xyz = 0^p 1^p$

$|xy| \leq p$

$y \neq \varepsilon$

$\Rightarrow y = 0^k$ for some choice of $k \in \{1 \ldots p\}$

$x y^i z \in A$ for all $i \in \mathbb{N}$.

in particular, for $i = 0$:

$x y^0 z = xz \in A$ ✓

but $xz = 0^{p-k} 1 \in A$ ✗

(contradiction)

Therefore A is not regular. □

Example:

$B = \{ w \in \{0,1\}^* : B \text{ doesn't have the same \# of 0's \& 1's} \}$

$0110 \in B$, $01001 \notin B$

Assume B is regular (pf by contra.)

$\Rightarrow \overline{B}$ is regular

$\Rightarrow \overline{B} \cap L(0^*1^*)$ is regular.

But, ← language of

$\overline{B} \cap L(0^*1^*) = A$, which is not regular.

↑ Pumping lemma

Example: $C = \{ w \in \{0,1\}^* : w = w^R \}$

↳ w is a palindrome

Claim: C is not regular.

Proof: Assume toward contradiction that C is regular.

By the PL, there is a pumping length p for C.

Let $w = 1^p 0 1^p$

we have $w \in C$ and $|w| = 2p+1 \geq p$.

Therefore, by the PL, it is possible to write $w = xyz$ such that

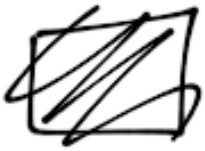
1. $y \neq \varepsilon$
2. $|xy| \leq p$
3. $xy^i z \in C$ for all $i \in \mathbb{N}$.

$$\left.\begin{array}{l} y \neq \varepsilon \\ |xy| \leq p_{p(p)} \\ xyz = 1 \text{ of } P \end{array}\right\} \Rightarrow y = 1^k \text{ for } 1 \leq k \leq p$$

$x y^i z \in C$ for all $i \in \mathbb{N}$, in particular

$$x y^2 z \in C$$

But $x y^2 z = 1^{p+1+k} 0 1^p \notin C$



Further Discussion of Regular Languages:

A, B are sets.

$\Rightarrow$ The symmetric difference of A and B is

$$A \Delta B = (A \setminus B) \cup (B \setminus A)$$

$$= (A \cup B) \setminus (A \cap B)$$

$\vee$.

Q.1: $A \subseteq \Sigma^*$ is regular, $B \subseteq \Sigma^*$ is it necessarily the case that $A \Delta B$ is regular?

Yes, B is finite $\Rightarrow$ B is regular

B is regular $\Rightarrow$ $\bar{B}$ is regular.

A regular $\Rightarrow \bar{A}$ regular

A regular, $\bar{B}$ regular $\Rightarrow A \cap \bar{B}$ regular

$\bar{A}$ regular, B regular $\Rightarrow \bar{A} \cap B$ regular

$A \cap \bar{B}$ regular, $\bar{A} \cap B$ regular $\Rightarrow (A \cap \bar{B}) \cup (\bar{A} \cap B)$ regular

A: $W = \{w_1, \dots w_n\} \subset A$ $X = \{x_1, \dots x_m\} \not\subset A$.

$$(A \setminus W) \cup X = A \Delta (W \cup X)$$

Q1b. $A \subseteq \Sigma^*$ is not regular. $B \subseteq \Sigma^*$ Finite.
$\Rightarrow A \Delta B$ non-regular.

Assume for Contradiction $A \Delta B$ regular.
$\Rightarrow (A \Delta B) \Delta B$ is regular
But $(A \Delta B) \Delta B = A$, which is not regular.

Q1C: $A \subseteq \Sigma^*$, $B \subseteq \Sigma^*$, (neither are regular)
Is it necessarily that $A \Delta B$ is non-regular?
False.
If $A = B = \{0^n 1^n, n \in \mathbb{N}\}$, $A \Delta B = \emptyset$, which is regular.

Q2: Supp. $A \subseteq \Sigma^*$ is regular.
Def: $A^R = \{w^R : w \in A\}$ is A^R necessarily regular?

Have a DFA "M" for A.
$M = (Q, \Sigma, \delta, q_0, F)$
Define NFA "N"
$N = (\quad , \Sigma, \eta, \quad , \{q_0\})$

$(Q \cup \{r_0\})$ $\quad r_0$

$\eta(r_0, \varepsilon) = F$
$\eta(q, \sigma) = \{p \in Q : \delta(p, \sigma) = q\}$
$\forall q \in Q, \sigma \in \Sigma$

A regular $\Leftrightarrow$ $\exists$ reg. exp'n R for A.

Supp S. is reg expr:

if $S = \emptyset$, $S = \varepsilon$, $S = \sigma$, $S^R = S$

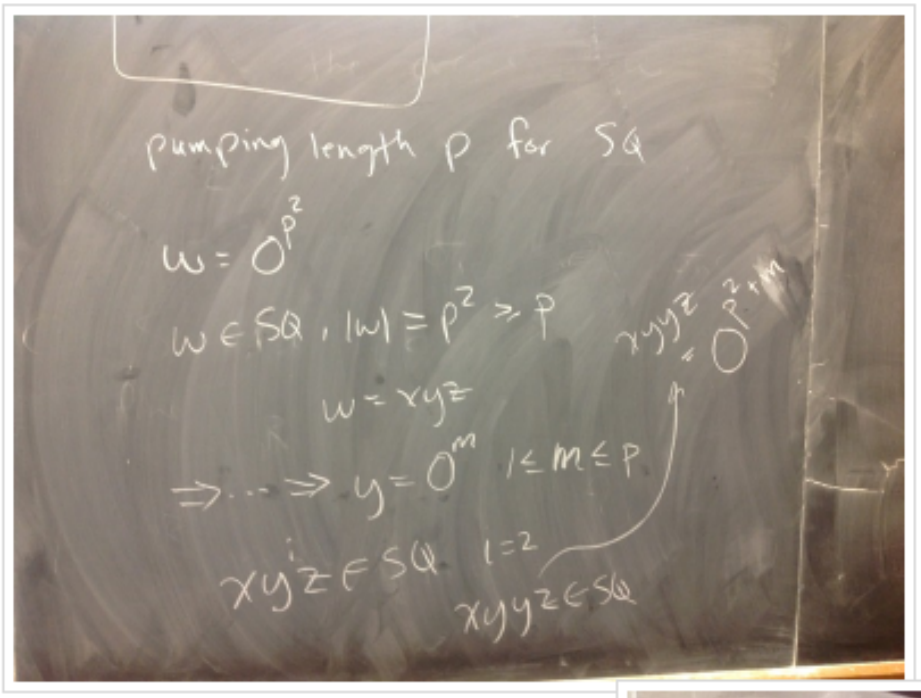
if $\begin{cases} S = S_1 \cup S_2 & \Rightarrow \quad S^R = (S_1^R \cup S_2^R) \\ S = S_1 S_2 & \Rightarrow \quad S^R = (S_2^R S_1^R) \\ S = S_1^* & \Rightarrow \quad S^R = (S_1^R)^* \end{cases}$

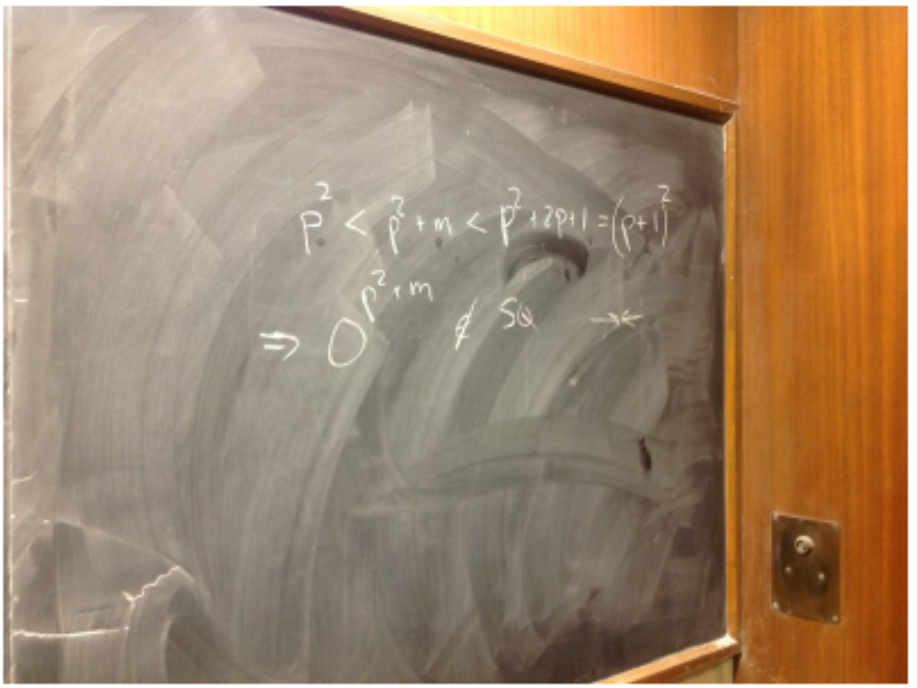
Exercise: do Q2 with reg. expr's

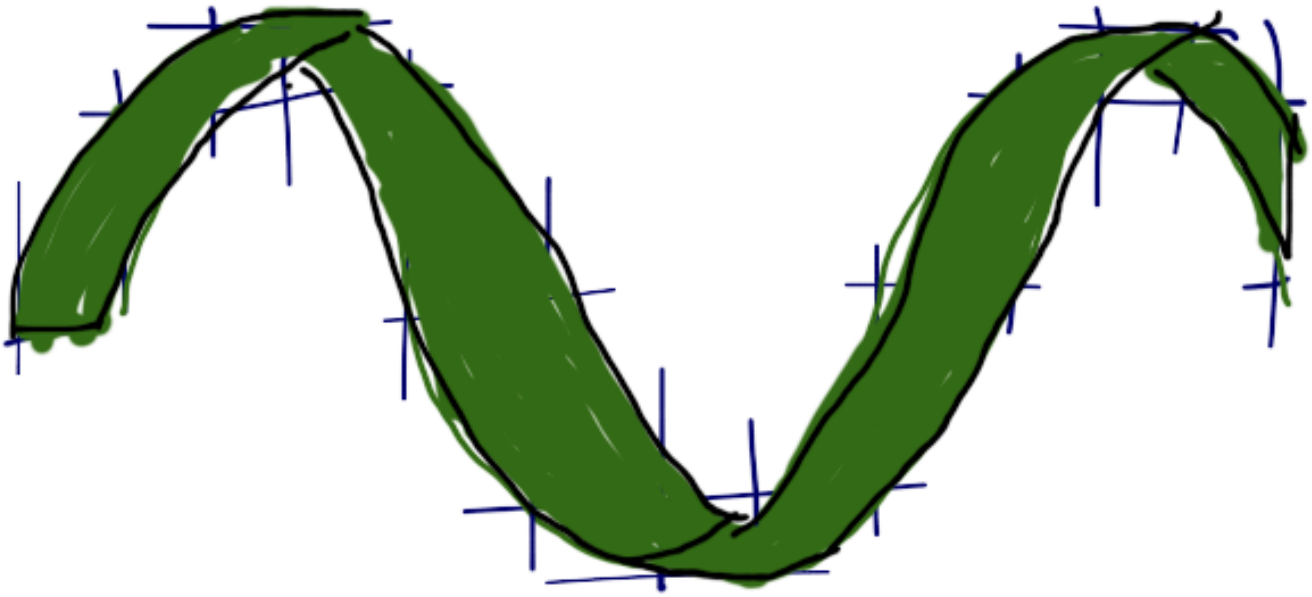
$\Sigma = \{0\}$

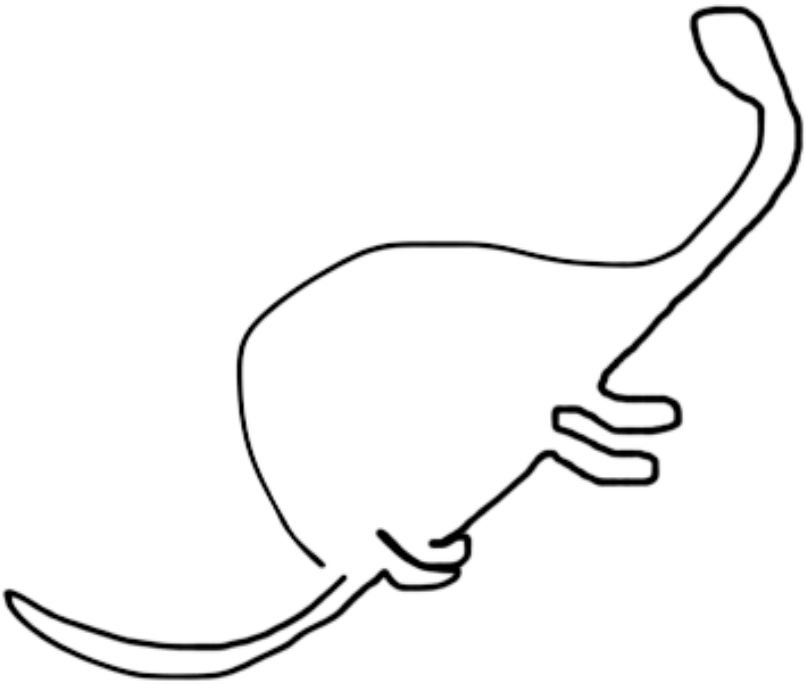
$SQ = \{0^n : n \text{ is a perfect sq}\}$

claim: SQ is not regular.









Context-Free Grammars:

A 4-tuple of:

$G = (V, \Sigma, R, S)$

where:

1. V is a finite, nonempty set of variables
2. Σ is an alphabet ($V \cap \Sigma = \emptyset$)
3. R is a finite set of rules of the form:

$$A \to w$$

$$A \in V, \quad w \in (V \cup \Sigma)^*$$

4. $S \in V$ is the start variable

ex:

$G = (V, \Sigma, R, S)$

$V = \{S\}$

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

R includes these 2 rules:

$S \to 0S1$

$S \to \varepsilon$

Every Grammar Generates a language:

$L(G) \subseteq \Sigma^*$

Informally:

1. write down start variable $S \rightarrow$ Current_str
2. loop:
   Pick any var in the current string with one of its expansions according to rules R of the C.F.L.
3. if you are left with $x \in \Sigma^*$, then x is generated by G.

Definition: Let $G=(V,\Sigma,R,S)$ be a CFG

1. For all $u,v,w\in(V\cup\Sigma)^*$ and $A\in V$, we have: $uAv \Rightarrow_G uwv$ if $A\to w\in R$
2. For $x,y\in(V\cup\Sigma)^*$, we have $x\Rightarrow_G^* y$ if there exists $k\geq 1$ and $w_1,\dots w_k\in(V\cup\Sigma)^*$ such that $x\Rightarrow_G w_1\Rightarrow_G w_2\Rightarrow_G \dots \Rightarrow_G w_k\Rightarrow_G y$
3. $L(G)=\{x\in\Sigma^*: S\Rightarrow_G^* x\}$

$S\Rightarrow_G w_1\Rightarrow_G w_2\dots\Rightarrow_G w_k=x$ is called a derivation of x

4. A Language $A\subseteq\Sigma^*$ is context-free iff there exists a CFG G such that $A=L(G)$

Ex: $\{x\in\{0,1\}^*: x=x^R\}$

$S\to 0S0$
$S\to 1S1$
$S\to 0$
$S\to 1$
$S\to\varepsilon$

Shorthand

$S\to 0S0\mid 1S1\mid 1\mid 0\mid\varepsilon$

Definition:

Supp: $M = (Q, \Sigma, \delta, q_0, F)$ is a DFA and $w \in \Sigma^*$. The DFA M accepts w iff

1. $w = \varepsilon$, and $q_0 = F$, or
2. $w = \sigma_1 \ldots \sigma_n$ for $n \geq 1$, $\sigma_1 \ldots \sigma_n \in \Sigma$ and there exist states $r_0 \ldots r_n \in Q$ such that $r_0 = q_0$, $r_n \in F$ and $\delta(r_k, \sigma_{k+1}) = r_{k+1}$ for all $k \in \{0 \ldots n-1\}$

An equivalent def:

- extend δ to operation $r \in \Sigma^*$.

Define:

$\delta^*: Q \times \Sigma^* \to Q$

as follows:

1. $\delta^*(q, \varepsilon) = q$
2. $\delta^*(q, w\sigma) = \delta(\delta^*(q, w), \sigma)$ for all $q \in Q$, $w \in \Sigma^*$, $\sigma \in \Sigma$

M accepts w iff $\delta^*(q_0, w) \in F$

Ex:

G:

$$S \to 0S1S \mid 1S0S \mid \varepsilon$$

$L(G) = ?$

$$A \stackrel{def}{=} \{x \in \{0,1\}^* : |x|_0 = |x|_1\}$$

$L(G) = A$ claim

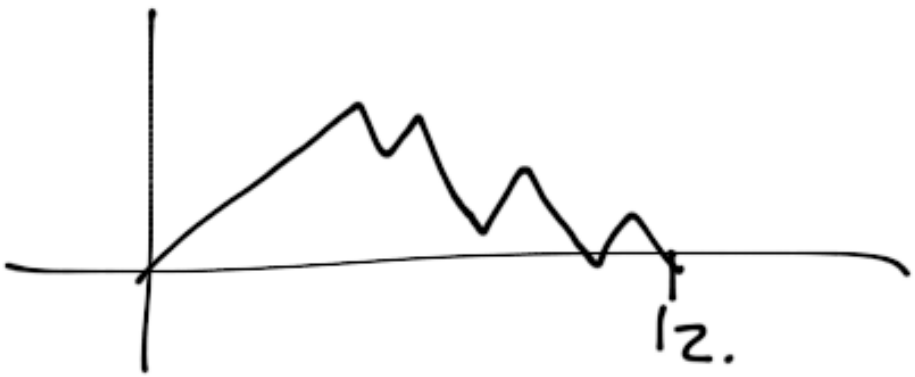
$L(G) \subseteq A$

clear from inspection of the grammar

$A \subseteq L(G)$

Supp $x \in A$

000101101



G: $S \to (S) \mid SS \mid \varepsilon$

$\Sigma = \{(,)\}$

$S \Rightarrow_G SS \Rightarrow_G (S)S \Rightarrow_G ((S))S \Rightarrow_G (())S$

$\Rightarrow_G (())(S) \Rightarrow_G (())() \Rightarrow (())() \in L(G)$

Likewise,

$\varepsilon, (), (()), ()(), ((())), \ldots \in L(G)$

$),)(, ((, \ldots \notin L(G)$

$L(G)$ is the language of balanced parens.

this is an ex. of Leftmost Derivation

we can always derive strings in $L(G)$ ($\forall$ grammars G) using a leftmost derivation.

$\Sigma = \{a, b, +, *, (,)\}$

$S \to S+S \mid S*S \mid (S) \mid a \mid b$ } "R"

$(a+b)*a+b$

$S \Rightarrow S*S \Rightarrow (S+S)*S \Rightarrow (a+S)*S \Rightarrow (a+b)*S$
$\Rightarrow (a+b)*S+S \Rightarrow (a+b)*a+S \Rightarrow (a+b)*a+b$

This grammar generates all formulas over a, b, +, * and parens

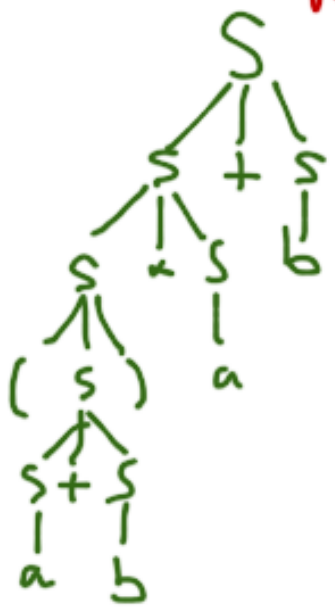
Parse Tree:

There is a one-to-one and onto (bijection?) b/w parse trees & leftmost derivations.

A CFG G is ambiguous if ∃ string s, s∈L(G) having 2 or more parse trees.

For many CFG applications we need unambig. grammars.

R_1, again (a+b)*a+b



Try again w/ R_2: (being unambiguous now)

$S \to T \mid S+T$
$T \to F \mid T*F$
$F \to I \mid (S)$
$I \to a \mid b$

unique parse tree (a+b)*a+b

Proving that various grammars are/aren't (un)ambig is very hard.

CFL w/no Unimbig CFG:

$\{0^n 1^m 0^k,\ n=m \text{ or } m=k\}$

$S \to A \mid B$
$A \to CD$
$C \to 0C1 \mid \varepsilon$
$D \to 0D \mid \varepsilon$
$B \to DE$
$E \to 1E0 \mid \varepsilon$

$\{0^n 1^m : n \neq m\}$

$S \to 0S1 \mid A \mid B$
$A \to 0A \mid 0$
$B \to B1 \mid 1$

$A = \{0^n 1^n : n \in \mathbb{N}\}$

Show A^* is context-free.

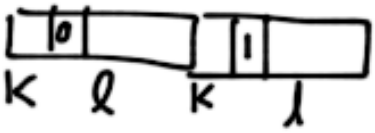
$S \to AS \mid \varepsilon$
$A \to 0A1 \mid \varepsilon$

A CFL A having no unambiguous CFG is said to be inherently ambiguous

$L_2 = \{u z v : u, v \in \{0,1\}^*, |u| = |v|, u \neq v^R\}$

$S \to 0A1 \mid 1A0 \mid 0S0 \mid 1S1$
$A \to XAX \mid z$
$X \to 0 \mid 1$

$L_3 = \{u v : u, v \in \{0,1\}^*\}$



$\to \quad x^k 0 x^{k+l} 1 x^l$

$S \to AB \mid BA$
$A \to XAX \mid 0$
$B \to XBX \mid 1$
$X \to 0 \mid 1$

Closure Props of CFLs:

Thm: Σ alphabet, $A, B \subseteq \Sigma^*$ are context-free languages.

$\Rightarrow$ These languages are also context-free:

$$A \cup B,\ AB,\ A^*$$

(The CFL's are closed under regular operations)

Pf: Let $G_A = (V_A, \Sigma, R_A, S_A)$
$G_B = (V_B, \Sigma, R_B, S_B)$

be CFGs, w/ $A = L(G_A)$ & $B = L(G_B)$

W/o loss of generality (wLOG), assume
$V_A \cap V_B = \emptyset$ and $S \notin V_A \cup V_B$

CFG for:

$A \cup B$:
$S \to S_A \mid S_B$

AB:
$S \to S_A S_B$

A^*
$S \to S_A S \mid \varepsilon$

$G = (V, \Sigma, R, S)$

$V = V_A \cup V_B \cup \{S\}$

$R = R_A \cup R_B \cup \{S \to S_A \mid S_B\}$

Thm: Supp Σ alphabet, $A \subseteq \Sigma^*$ is a regular language.

$\Rightarrow A$ is a context-free language.

Pf. 1 (using regular expr)

Let R be a regular expr. for A.

We will define a CFG G s.t. $L(G) = L(R)$

$\square$

Left for FB interview

Proving some languages are not context free

Pumping Lemma For CFLs:

$\forall \Sigma$ and $\forall A \subseteq \Sigma^*$

$\exists$ pumping length $p \geq 1$ for A where $\forall w \in A$: $|w| \geq p$ and it is possible to write $w = uvwxyz$ for $u, v, w, x, y, z \in \Sigma^*$ such that:

1. $v y \neq \varepsilon$
2. $|vxy| \leq p$
3. $uv^i x y^i z \in A, \quad \forall i \in \mathbb{N}$

Pf:

Given that A is Context-Free, there exists a CFG G such that $A = L(G)$

Assume WLOG that G is in chomsky normal flow.

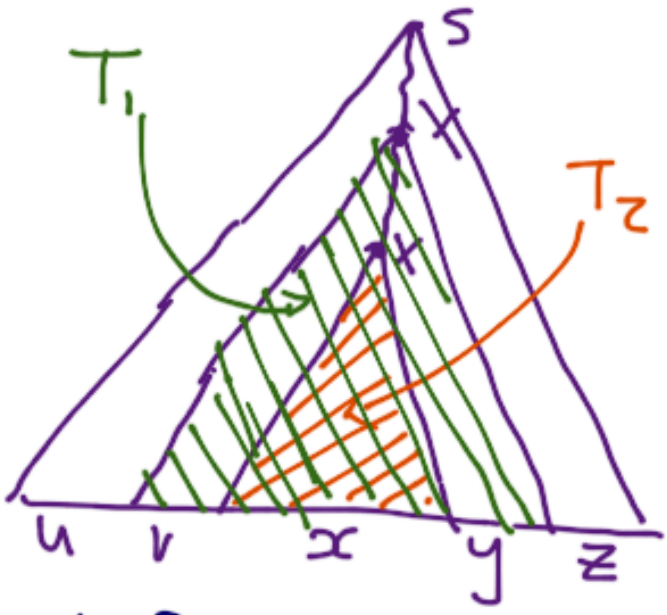
let m be the number of variables in G, and let $p = 2^m$

Suppose $w \in A$ $|w| \geq p = 2^m$, G is in Chomsky normal Flow, so any parse tree for w has $2|w|-1$ internal (variable) nodes

$$2|w|-1 \geq 2 \cdot 2^m - 1 > 2^m$$

at least 2^m nodes.

There must be a path in the tree root to a leaf with at least $m+1$ variable nodes.



we define u, v, x, y, z as the picture suggests

1. $vy \notin E$, otherwise we could build a strictly smaller parse tree for w... impossible in CNF.

3. Continue this way to obtain a parse tree for $uv^i xy^i z$, $i \in \mathbb{N}$.

2. we want to choose T_1 as small as possible to prove

$$|vxy| \leq p$$

Using the pumping Lemma For CFG-s (an ex.)

$$A = \{0^n 1^n 2^n : n \in \mathbb{N}\}$$
$$\Sigma = \{0,1,2\}$$

Claim: A is not Context-free

Proof: Assume towards contradiction that A is context-free.

By the PL for CFLs, there exists a p for A.

Let $w = 0^p 1^p 2^p$. We have $w \in A$ and $|w| = 3p \geq p$. So there must exist $u, v, x, y, z \in \Sigma^*$ such that

- $w = uvxyz$
- $vy \neq \varepsilon$
- $|vxy| \leq p$
- $uv^i xy^i z \in A$ for all $i \in \mathbb{N}$.

$w = 0^p 1^p 2^p = uvxyz$

$|vxy| \leq p$, so it is impossible for all 3 symbols 0, 1, 2 to appear within vxy

$vy \neq \varepsilon$, so at least one of the 3 symbols must appear within vy.

So, taking $i=0$, we know that uv^0xy^0z must contain strictly fewer of one of the symbols than another.

This conflicts the statement that $uv^0xy^0z \in A$

$\Rightarrow\!\!\!\!\times$ Therefore A is not context-free.

Ex²: $B = \{r2s : r, s \in \{0,1\}^*, r \text{ is a substring of } s\}$

Claim B is not Context-Free

Proof: Assume towards contra. that B is context-Free.

By the PL for CFLs, there must be a pumping length $p \geq 1$ for B.

Let $w = 0^p 1^p 2 0^p 1^p$, $w \in B$, and $|w| > p$

So it is possible to write $w = uvxyz$ so that 1,2,3 from the CFG PL holds

There are these possibilities for the location of the 2 in $uvxyz$.

Case 1: z lies within u

Consider $i=0$

$uv^0xy^0z = uxz \in B$.

we have $vy \neq \varepsilon$, so

$uxz = 0^p 1^p 2 0^{p-k} 1^{p-l}$, $k+l \geq 1$

So this is not a string in B, so we have a contradiction for this case.

Case 2: the z lies within v or y.

Consider $i=2$

uv^2xy^2z has 2 z's, so it is invalid.

Pumping Lemma For CFLs:

$$(\forall A \subseteq \Sigma^* : A \text{ is a CFL})(\exists p \geq 1)(\forall w \in A : |w| \geq p,$$
$$w = uvxyz :: vy \neq \varepsilon \wedge |vxy| \leq p \wedge (\forall i \in \mathbb{N}$$
$$uv^i x y^i z \in A))$$

$B = \{ r2s : r, s \in \{0,1\}^*, r \text{ is a substring of } s \}$

Claim: B is not context-free.

Pf: Towards contradiction, assume B is context free.... pumping length P

let $w = 0^p 1^p 2 0^p 1^p$, $w = uvxyz$..

Case 1: $2 \in u$ [yesterday]

Case 2: $2 \in v$ or $2 \in y$ [yesterday]

Case 3a: $z \in x$

$uvxyz = 0^p 1^p 2 0^p 1^p$

z

We have $|vxy| \leq p$

It must be that $v = 1^k$ and $y = 0^l$

$(k+l \geq 1)$

3a: $k > 0$ consider $i = 2$

$uv^2xy^2z = 0^p 1^{p+k} 2 0^{p+l} 1^p \notin B$

3b: $l > 0$ consider $i = 0$

$uv^0xy^0z = uxz = 0^p 1^{p-k} 2 0^{p-l} 1^p \notin B$

Case 4: $z \in z$

Consider $i = 2$

$uv^2xy^2z \notin B$ because it has the form $r 2 0^p 1^p$ where $|r| > 2p$.

$\Rightarrow B$ is not context free

CFLs are Not closed under Intersection.

Ex: $A = \{0^n 1^n 2^k : n, k \in \mathbb{N}\}$

$S \Rightarrow XY$

$X \rightarrow 0X1 \mid \varepsilon$

$Y \rightarrow 2Y \mid \varepsilon$

A is context-free.

$B = \{0^k 1^n 2^n : n, k \in \mathbb{N}\}$

B is context-free.

$A \cap B = \{0^n 1^n 2^n : n \in \mathbb{N}\}$

which is not context-free

CFLs are not closed under complement.

Follows DeMorgan's Laws

Ex: $C = \{0^n 1^n 2^n : n \in \mathbb{N}\}$

prove $\bar{C}$ is context-Free

CFG for $\bar{C}$:

$S \to X Y_2 \mid Y_0 W \mid U1U \mid U2U \mid U2U$

$X \to 0X1 \mid Y_0 0 \mid Y_1 1$

$Y_0 = 0Y_0 \mid \varepsilon$

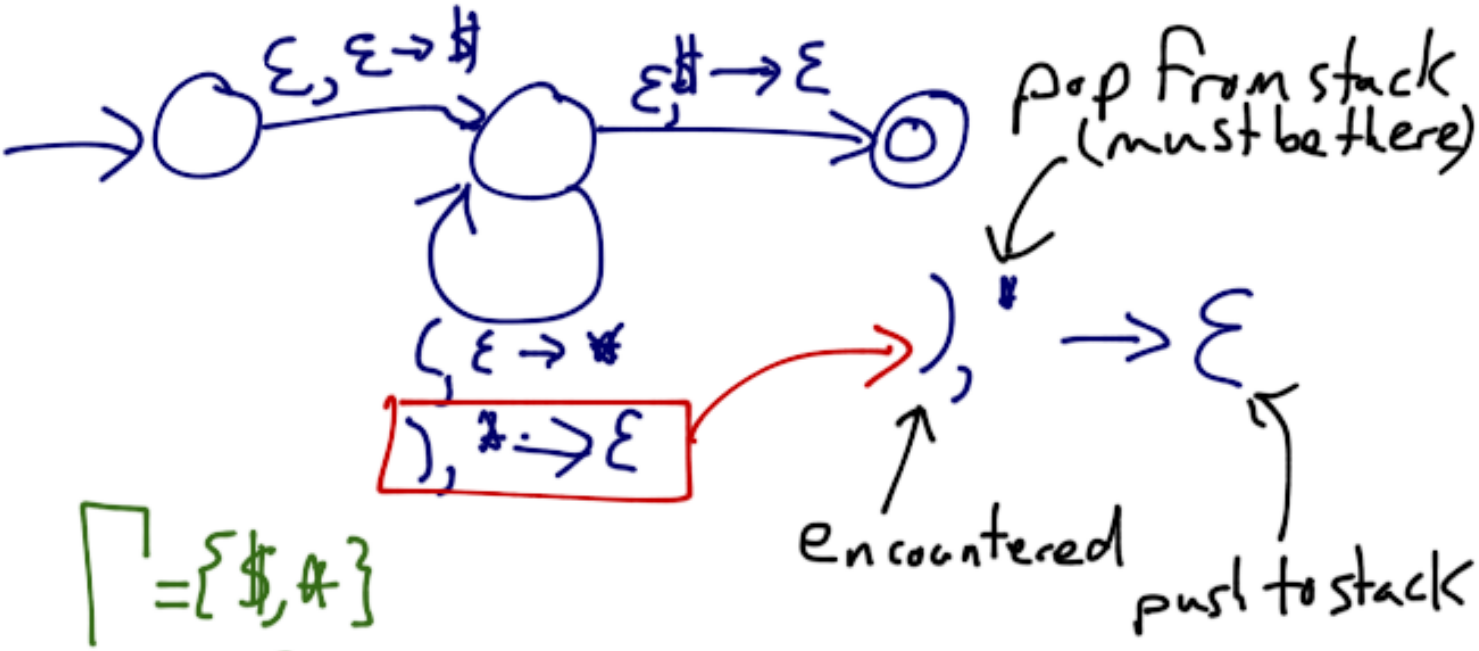
$Y_1 = 1Y_1 \mid \varepsilon$

$Y_2 = 2Y_2 \mid \varepsilon$

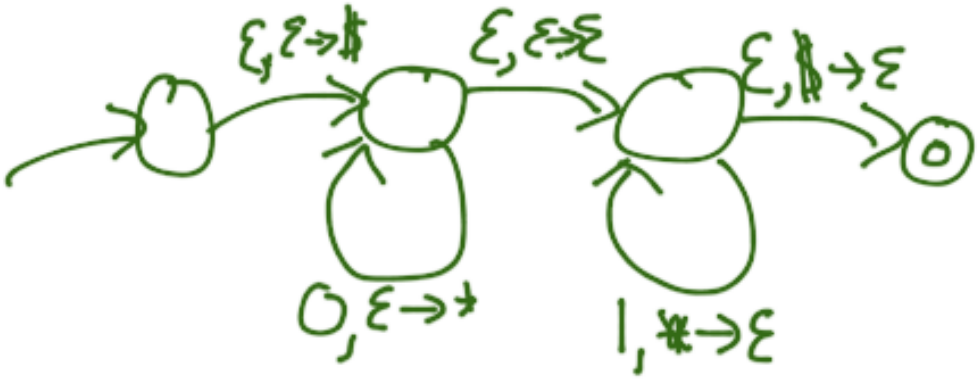
$U \to 0U \mid 1U \mid 2U \mid \varepsilon$

$W \to 1W2 \mid Y_1 \mid Y_2$

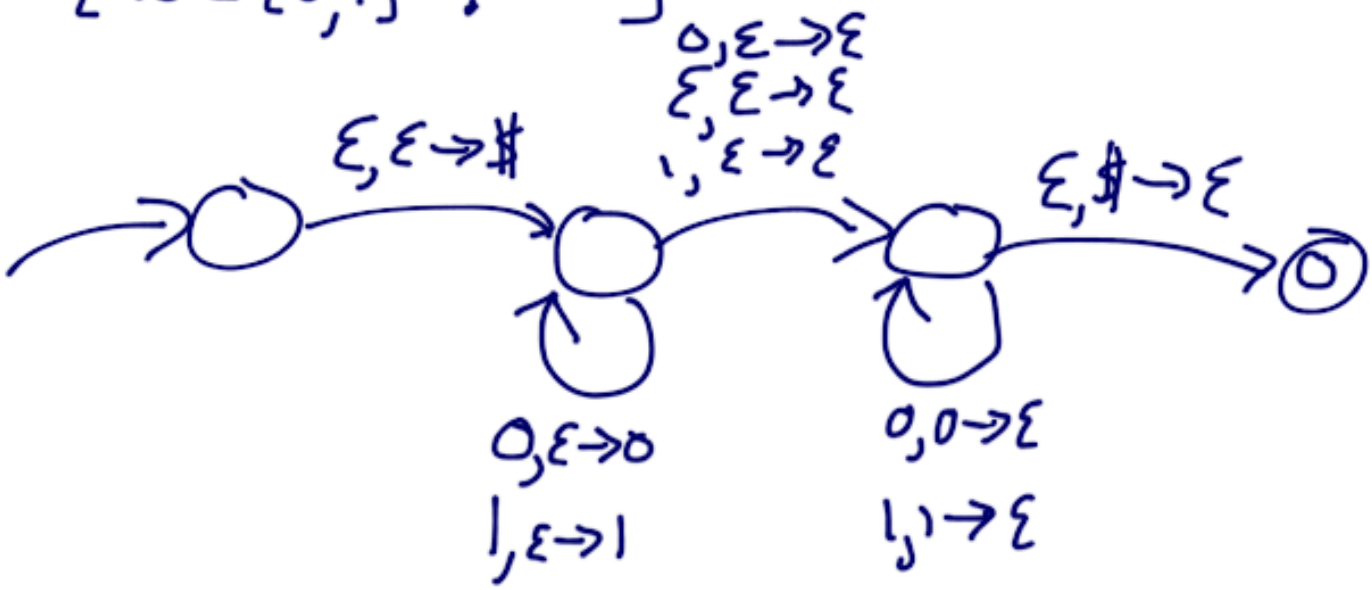
Pushdown Automatons: NFA's with 1 stack



$\{0^n 1^n : n \in \mathbb{N}\}$



$\{w \in \{0,1\}^* : w = w^R\}$



Thm Supp $A \subseteq \Sigma^*$

A is CF iff $A = L(P)$ for some P


NOT ON MIDTERM

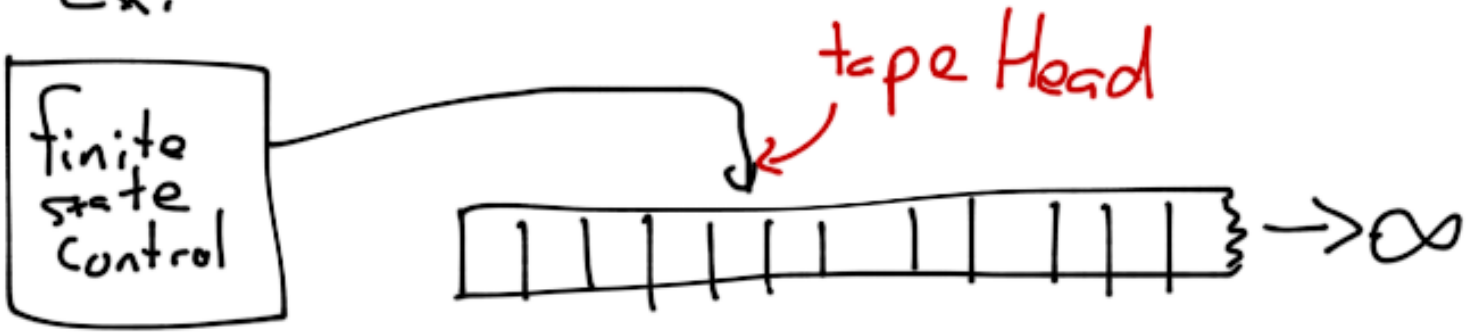
Turing Machines:

Church-Turing Thesis.

∀ Functions that can be computed by mechanical means can be computed by a Turing machine.

Allan Turing (1912-1954)
- Turing machine proposed in 1936.
- Simple Mathematical model for general computations.

Ex:



1. Finite state ctl:
   Stores one of a finite set Q of states at each instant.
2. Tape: Segmented read/write tape that stores alphabet chars.
3. Tape Head. Scans one tape square @ each moment. Can move both directions.

Formal Definition:

A deterministic Turing Machine (DTM) is a 7-tuple:

$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$

(gamma)

- Q: states
- Σ: input alphabet
- Γ: tape alphabet
- δ: transition f'n
- q_0, q_{acc}, q_{rej}: start, accept, reject states,

Σ input alphabet

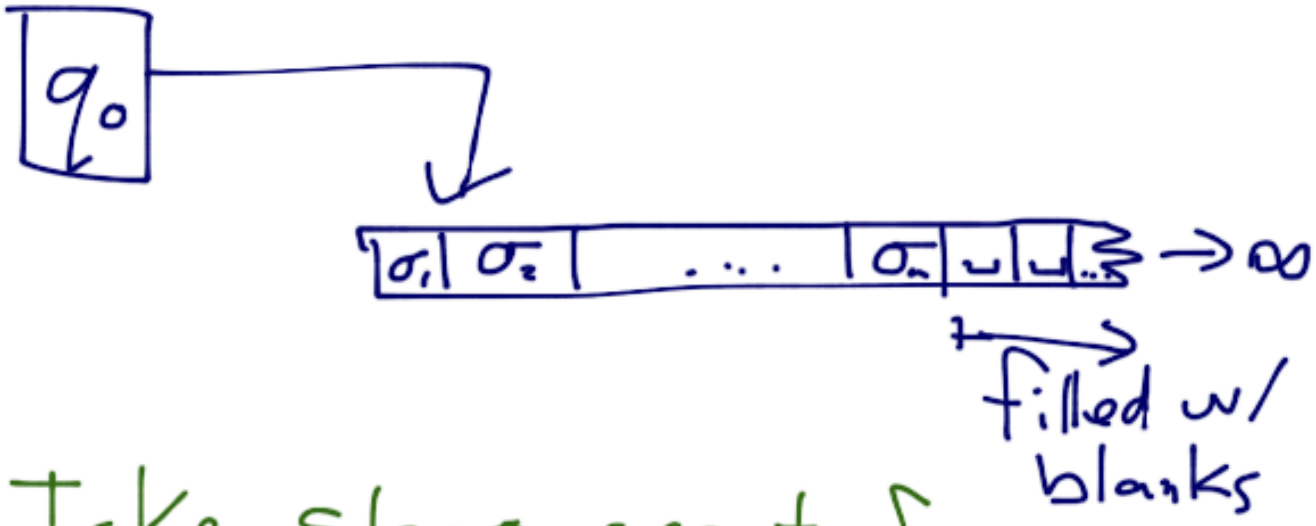
Γ tape alphabet

$\Sigma \subseteq \Gamma$

blank symbol (must be here) → $\sqcup \in \Gamma \setminus \Sigma$

If we wanted to run M on an input $x = \sigma_1 \sigma_2 \sigma_3 \ldots \sigma_n \in \Sigma^*$:

Start M like this:



Take steps acc. to δ:

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$$

↑ directions.

i.e:

$$\delta(q, \sigma) = (r, \tau, D)$$

means:

if state $= q$ & σ is read:

then:

- change state to r
- write τ on scanned tape square
- move head 1 step in dir D.

3poss. Outcomes of a Compilation:

1. Accept
2. Reject
3. Run forever.

A configuration of a DTM M is a description of everything there is to describe about M at some moment.

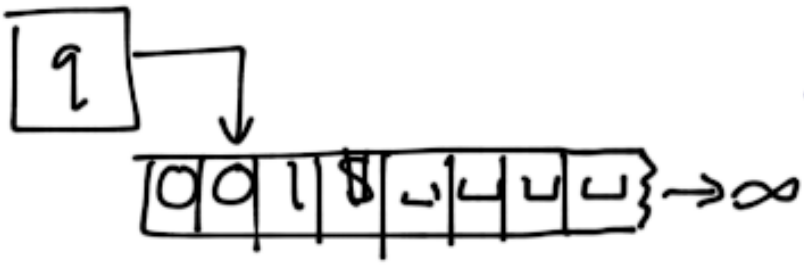
1. State
2. Contents of tape
3. position of tape head

Short hand of config:

$uqv : (q \in Q, u, v \in \Gamma^*)$

means: tape contains uv (followed by $\infty \times \sqcup$)

- State is q
- tapehead is scanning the 1st symbol of v.



$\Leftrightarrow 0q0|\$$

If we fix Σ, Γ, how many DTMs of the form $M=(Q,\Sigma,\Gamma,\ldots)$ are there?

Countably Many.

Input: $x \in \Sigma^*$

Start config: $q_0 x$

we can define yields as

$$C \vdash C'$$

means config C goes to C' in one step.

[$C \vdash^* C'$ is an arbitrary number (≥ 0) of steps]

$L(M) = \{w \in \Sigma^* : w \text{ is accepted by } M\}$

A lang $A \subseteq \Sigma^*$ is Turing-Recognizable iff $A = L(M)$

For some DTM M.

A language $A \subseteq \Sigma^*$ is decidable if $\exists$ a DTM M such that.

1. $A = L(M)$
2. M never runs forever

If M enters either of the states q_{acc} or q_{rej}, it Halts. (i.e. stops running)

i.e $\{0^n 1 \sqcup^n 10^n : n \in \mathbb{N}\}$ is decidable.

Argument:

Augmented Turing machine
(i.e Turing Machine + feature)
is still a turing machine, because we
can simulate a turing machine with it.