Chapt. 3 Properties of Reg. Sets

We study the following topics:

(1) Pumping lemma for reg. sets & applications

(2) Closure properties of reg. sets w.r.t. certain operations

(3) Decision algorithms for reg. sets

3.1. The Pumping Lemma for Reg. Sets.

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA.
Let $n := \text{Card}(Q)$. Let $L = L(M)$.
Consider an input string z

$$z = z_1 z_2 \dots z_m$$

of length $|z| = m \geq n$.

The computation of M on z has the form:

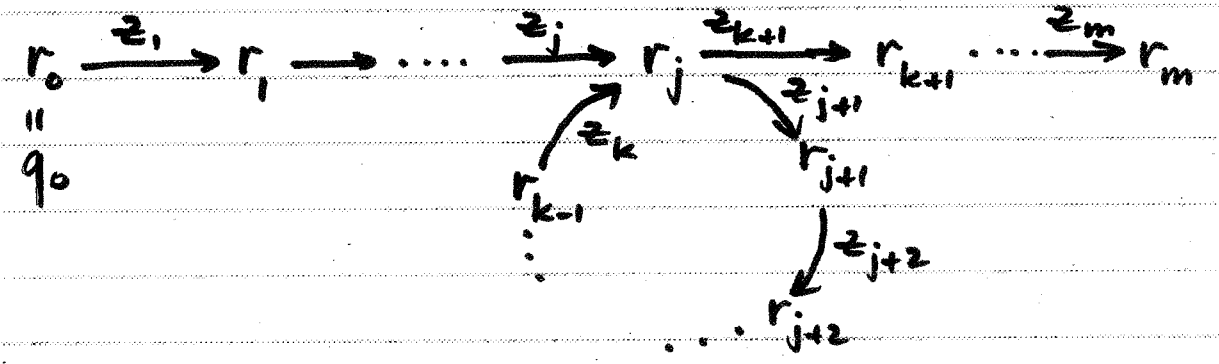
$$r_0 = q_0 \xrightarrow{z_1} r_1 \xrightarrow{z_2} r_2 \cdots \xrightarrow{z_m} r_m$$

where $r_0, r_1, \dots, r_m \in Q$.

Since $m \geq n$, there exist j, k with $0 \leq j < k \leq n$ s.t.

$$r_j = r_k$$

Thus, the computation path has the form:



Hence, $\forall i = 0, 1, \ldots$

$$\delta(r_0, z_1 \ldots z_j (z_{j+1} \ldots z_k)^i z_{k+1} \ldots z_m) = r_m$$

Therefore, if $r_m \in F$, then

$$\forall i \geq 0 : \underbrace{z_1 \ldots z_j}_{u} (\underbrace{z_{j+1} \ldots z_k}_{v})^i \underbrace{z_{k+1} \ldots z_m}_{w} \in L$$

Pumping Lemma for reg. sets.

Let $L \subseteq \Sigma^*$ be a reg. set. Then there exists a constant $n > 0$ s.t. for any $z \in L$ with $|z| \geq n$, z can be written as $z = uvw$ satisfying

(1) $|uv| \leq n$ (2) $|v| \geq 1$

(3) $\forall i \geq 0 \quad uv^i w \in L$

Application of Pumping Lemma

The Pumping Lemma for reg. sets has the form:

$$L \text{ is reg.} \Longrightarrow P$$

where

$$P \equiv \exists n>0\ \forall z\in L\Big[\,|z|\geq n \Longrightarrow (\exists u\,\exists v\,\exists w[(z=uvw \wedge |uv|\leq n \wedge |v|\geq 1)\wedge \forall i\geq 0\,(uv^iw\in L)])\Big]$$

Its contrapositive is

$$\neg P \Longrightarrow L \text{ is } \underline{\text{not}} \text{ reg.}$$

Suppose that we can prove for a given language L:

$$Q \equiv \forall n>0\ \exists z\in L\Big[\,|z|\geq n \wedge (\forall u\,\forall v\,\forall w[z=uvw \wedge |uv|\leq n \wedge |v|\geq 1 \Longrightarrow \exists i_0\geq 0\,(uv^{i_0}w\notin L)])\Big]$$

Then $Q \Rightarrow \neg P$

Hence $Q \Rightarrow L$ is not reg.

<u>Remark</u>. Q means: for any integer $n>0$ there is a string $z\in L$ with $|z|\geq n$ s.t. for any factorization of z as $z=uvw$ with $|uv|\leq n$, $|v|\geq 1$, there is i_0 s.t. $uv^{i_0}w\notin L$

Ex. **Claim.** $L = \{0^{k^2} \mid k \geq 1\}$ is not reg.

Pf.

(1) Let $n > 0$ be an arbitrary const.

(2) Let $z = 0^{n^2}$. Then $z \in L \wedge |z| \geq n$

(3) Consider an arbitrary factorization of z as $z = uvw$ with $|uv| \leq n$, $|v| \geq 1$

(4) Let $i_0 = 2$. Then

$$z' = uv^2w = 0^{n^2+l}$$

where $1 \leq l = |v| \leq n$ (since $|uv| \leq n$).

Now $n^2 < |z'| = n^2 + l \leq n^2 + n < (n+1)^2$

That is $|z'|$ is between two consecutive perfect squares: n^2 and $(n+1)^2$.

Thus, $z' \notin L$

(5) Therefore, Q is satisfied.

We conclude that L is not reg. ⊠

Thus to show that a given lang. L is not regular we follow the 5 steps:

(1) Let $n > 0$ be an arbitrary const.

(2) Choose a string z s.t. $z \in L \wedge |z| \geq n$

(3) Consider an arbitrary factorization of z as $z = uvw$ with $|uv| \leq n$ and $|v| \geq 1$

(4) Find an integer $i_0 \geq 0$: $uv^{i_0}w \notin L$

(5) Conclude that L is not regular since it doesn't satisfy P.L. □

Ex: **Claim.** $L = \{ 0^k 1^k \mid k \geq 0 \}$ is not reg.

Pf: (1) Let $n > 0$ be an arbitrary const.

(2) Let $z = 0^n 1^n$. Then $z \in L \wedge |z| \geq n$.

(3) Consider an arbitrary factorization of z as $z = uvw$ with $|uv| \leq n$ and $|v| \geq 1$.

(4) Let $i_0 = 2$. Then $z' = uv^2w = 0^{n+l} 1^n$ where $1 \leq l = |v| \leq n$.
Since $\#_0(z') = n + l \neq n = \#_1(z')$, $z' \notin L$.
(Choosing $i_0 = 0$ would work too)

(5) We conclude that L is not reg. □

Ex: **Claim.** $L = \{0^i 1^j \mid i < j\}$ is not reg.

Pf. (1) Let $n > 0$ be an arbitrary const.

(2) Let $z = 0^n 1^{n+1}$. Then $z \in L \wedge |z| \geq n$.

(3) Consider an arbitrary factorization of z as $z = uvw$ with $|uv| \leq n$ and $|v| \geq 1$.

(4) Let $i_0 = 2$. Then $z' = uv^2w = 0^{n+\ell} 1^{n+1}$ where $1 \leq |v| = \ell \leq n$.
Clearly, $z' = 0^{n+1+\ell'} 1^{n+1}$, $0 \leq \ell' \leq n-1$
Since $\#_0(z') \geq \#_1(z')$, $z' \notin L$

(5) We conclude that L is not reg. □

Remark. In the above ex. choosing $i_0 = 0$ does not work!

Ex: **Claim.** $L = \{0^i 1^j \mid i > j\}$ is not reg.

Pf. (1) Let $n > 0$ be an arbitrary const.

(2) Let $z = 0^{n+1} 1^n$. Then $z \in L \wedge |z| \geq n$

(3) Consider an arbitrary factorization of z as $z = uvw$ with $|uv| \leq n \wedge |v| \geq 1$

(4) Let $i_0 = 0$. Then $z' = uv^0w = 0^{n+1-\ell} 1^n$ where $1 \leq \ell = |v| \leq n$. Clearly, $z' = 0^{n-\ell'} 1^n$ where $0 \leq \ell' \leq n-1$. Hence, $z' \notin L$

(5) We conclude that L is not reg. □

Ex. **Claim.** $L = \{0^p \mid p \text{ is prime}\}$ is not reg.

Pf: (1) Let $n > 0$ be an arbitrary const.

(2) Let $z = 0^p$, where $p \geq n$ is prime.
Then $z \in L$ and $|z| \geq n$.

(3) Consider an arbitrary factorization of z as $z = uvw$ with $|uv| \leq n \wedge |v| \geq 1$.

(4) [Let $i_0 = 2$. Then $z' = uv^2w = 0^{p+l}$
where $1 \leq l = |v| \leq n$.
We wish to show that $z' \notin L$,
i.e., $|z'|$ is a composite number.
Q. Can we show that $p+l$ is composite?
For ex., if $n = 5$, $p = 7$ and $l = 4$,
then $p + l = 7 + 4 = 11$ is again prime.
$\Rightarrow$ $i_0 = 2$ does not work!]

Let $i_0 = p+1$. Then $z' = uv^{p+1}w$.
We have: $|z'| = |uvw| + |v^p|$
$= p + p \cdot |v| = p(1+|v|)$
is a composite number

Thus, $z' \notin L$

(5) We conclude that L is not reg. □

Ex: Claim. $L = \{0^i 1^j \mid i \neq j\}$ is not reg.

Pf. (1) Let $n > 0$ be an arbitrary const

(2) [Let $z = 0^n 1^{n+2}$. Then $z \in L$ and $|z| \geq n$.
Consider an arbitrary factorization of z as $z = uvw$ with $|uv| \leq n \wedge |v| \geq 1$.
Let $i_0 = 2$. Then $z' = uv^2w = 0^{n+l} 1^{n+2}$,
where $1 \leq l = |v| \leq n$.

Q. Can we show $n+l = n+2$?

No: since $l = 1$ still sat. $1 \leq l \leq n$.
Thus this choice of z does not work]

Let $z = 0^n 1^{n+n!}$. Then $z \in L \wedge |z| \geq n$.

(3) Consider an arbitrary factorization of z as $z = uvw$ with $|uv| \leq n \wedge |v| \geq 1$.

(4) Let $i_0 = \frac{n!}{l} + 1$, where $l = |v|$.

Then $z' = uv^{\frac{n!}{l}+1}w = uvv^{\frac{n!}{l}}w$

$= 0^n (0^{|v|})^{\frac{n!}{l}} 1^{n+n!}$

$= 0^{n+n!} 1^{n+n!}$

Clearly, $z' \notin L$

(5) We conclude that L is not reg. □

How to choose $z = 0^n 1^{n+?}$
s.t. we can pump 0s and the resulting string $z' \notin L$.
Let $i_0 = 1+\lambda$, where λ is to be def.
Consider $z' = uv^{i_0}w$, $1 \leq |v| = \ell \leq n$

$$= 0^{n-\ell}(\underbrace{0^{\ell}}_{v})^{i_0} 1^{n+?}$$

$$= 0^{n-\ell}(0^{\ell})^{1+\lambda} 1^{n+?}$$

$$= \underline{0^{n-\ell}\ 0^{\ell}}\, 0^{\ell\cdot\lambda} 1^{n+?}$$

$$= 0^{n} 0^{\ell\cdot\lambda} 1^{n+?}$$

Must choose λ s.t. ℓ got cancelled.
Thus, λ must be of form $\lambda = \frac{\square}{\ell}$
That means $\square$ must have ℓ as a factor for any $1 \leq \ell \leq n$: $\square = n!$
Hence choose $\lambda = \frac{n!}{\ell}$.
Then $z' = 0^n 0^{\ell\cdot\frac{n!}{\ell}} 1^{n+n!}$

$$= 0^{n+n!} 1^{n+n!}$$

Thus, choose $z = 0^n 1^{n+n!}$
and $i_0 = 1 + \frac{n!}{\ell}$

3.2. Closure Properties of Reg. Sets

We study the closure properties of reg. sets under the operations $\cup, \cap, ^-$, $\cdot$ and *.

Proposition. Regular sets are closed under the regular operations $\cup, \cdot$ and *. That is if L_1 and L_2 are regular, then so are $L_1 \cup L_2$, $L_1 \cdot L_2$ and L_1^*.

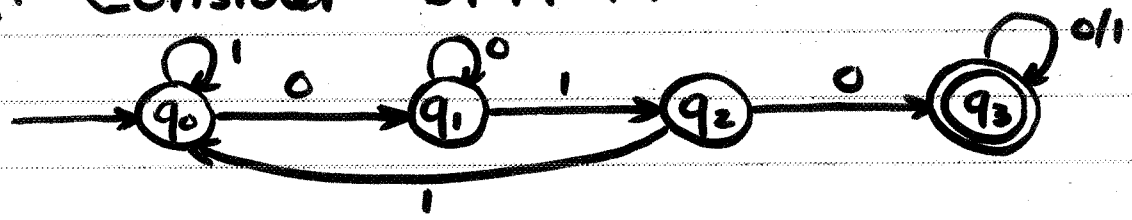
Pf. This is obvious since the reg. expr. are def. recursively from $+, \cdot$ and *. Also note that in showing how to convert a given reg. expr. to an equiv. NFA we showed how to construct for given NFAs N_1 and N_2 an NFA N that accepts $L(N_1) \cup L(N_2)$, or $L(N_1) \cdot L(N_2)$, or $L(N_1)^*$. □

Proposition. Regular sets are closed under complement, i.e., if $L \subseteq \Sigma^*$ is reg, then so is $\overline{L} = \Sigma^* - L$.

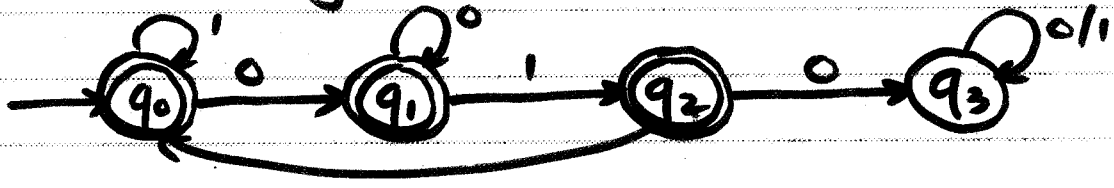
Pf. Let $L \subseteq \Sigma^*$ be a reg. set.
Let $L = L(M)$ for some DFA
$M = (Q, \Sigma, \delta, q_0, F)$.
Then $\overline{L} = L(\underbrace{(Q, \Sigma, \delta, q_0, Q-F)}_{\overline{M}})$.

That is $\overline{M}$ is obtained from M by making final states F of M nonfinal, and vice versa. □

Ex. Consider DFA M



that accepts bin. strings containing 010 as substring. Then $\overline{M}$



accepts bin. strings that do not contain 010 as substring.

Proposition. Regular sets are closed under intersection. That is if L_1 and L_2 are regular, then so is $L_1 \cap L_2$

Pf. Consider the following equality

$$L_1 \cap L_2 = \overline{\overline{L_1 \cap L_2}} = \overline{\overline{L_1} \cup \overline{L_2}}.$$

If L_1 and L_2 are regular, then
$\overline{L_1}$ and $\overline{L_2}$ are regular
(since reg. sets are closed under $^-$),
and hence
$\overline{L_1} \cup \overline{L_2}$ is regular
(since reg. sets are closed under $\cup$)

Therefore
$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$ is regular. □

A constructive proof.

Let L_1 and L_2 be accepted by DFAs
$M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ and
$M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$, resp.

We construct from M_1 and M_2 a DFA M for $L_1 \cap L_2$.

Idea: Simulate M_1 and M_2 in parallel and accept if both M_1 and M_2 accept.

The states of M is

$$Q = Q_1 \times Q_2 ,$$

initial state is (q_1, q_2) and

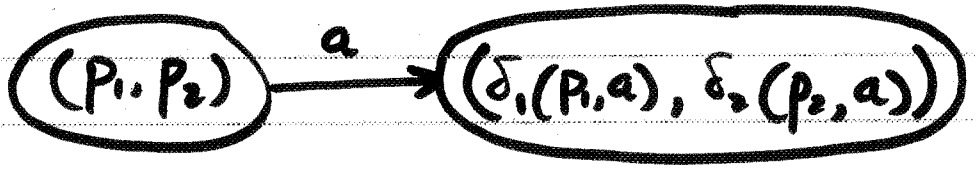
final states are $(s_1, s_2) \in F_1 \times F_2$.

To simulate M_1 and M_2 simultaneously, the trans. fctn of M is

$$\delta : (Q_1 \times Q_2) \times \Sigma \longrightarrow Q_1 \times Q_2$$

where

$$\delta((p_1, p_2), a) = (\delta_1(p_1, a), \delta_2(p_2, a))$$

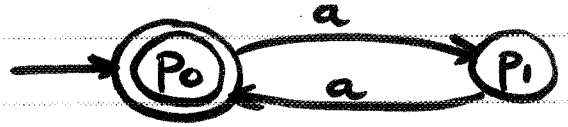


So, $M = (Q_1 \times Q_2, \Sigma, \delta, (q_1, q_2), F_1 \times F_2)$

Then $L(M) = L_1 \cap L_2$ □

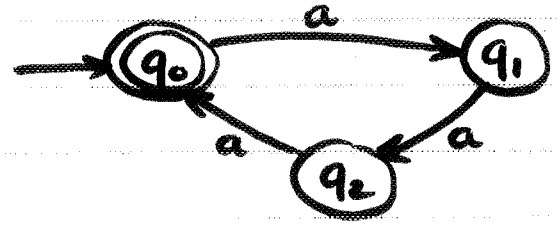
Ex. $L_1 = \{ a^{2k} \mid k \geq 0 \}$

M_1



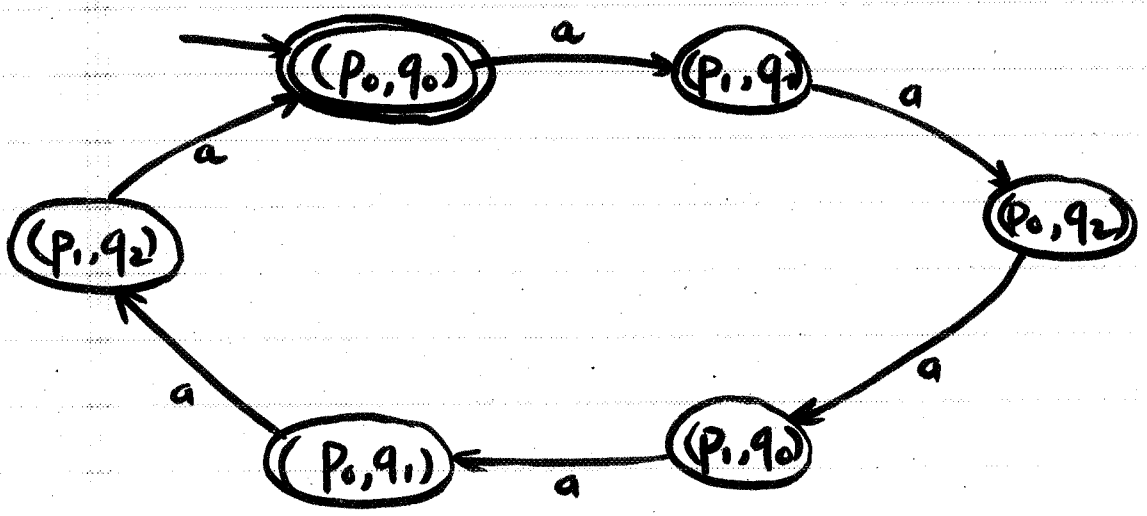
$L_2 = \{ a^{3k} \mid k \geq 0 \}$

M_2



$L_1 \cap L_2 = \{ a^{6k} \mid k \geq 0 \}$

The DFA M for $L_1 \cap L_2$ as constr. above is:



Note that M counts modulo 6

whereas M_1 ——— 2

and M_2 ——— 3 □

An application of closure properties.

Consider the

Claim. $L = \{ 0^i 1^j \mid i \neq j \}$ is not reg.

Based on the fact that $L' = \{ 0^i 1^i \mid i \geq 0 \}$ is not reg. we can give a simpler proof for this Claim:

Suppose by way of contradiction that L were reg. Then

$$\overline{L} \cap 0^* 1^* = \{ 0^i 1^i \mid i \geq 0 \} = L'$$

would be reg. ↝ a contradiction □

Ex: **Claim.** Let $L \subseteq \Sigma^*$ be a reg. set and $w \in \Sigma^*$ be a string. Then the set L' of strings in L that contain w as a substring and are of even length is regular.

Pf. Let L_1 be the set of strings over Σ containing w as substring. Then L_1 is denoted by $\Sigma^* w \Sigma^*$. Hence, L_1 is reg.

Let L_2 be the set of even length strings over Σ. Then $L_2 = (\Sigma^2)^*$. Thus, L_2 is also regular.

Since $L' = L \cap L_1 \cap L_2$

and reg. sets are closed under $\cap$, it follows that L' is also reg. □

Ex: **Claim.** If $L_1, L_2 \subseteq \Sigma^*$ are regular, then so is $L_1 - L_2 = \{ w \mid w \in L_1 \wedge w \notin L_2 \}$

Pf. Observe that

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

Since reg. sets are closed under $\overline{\ }$ and $\cap$, $L_1 - L_2$ is reg. □

3.3. Decision Algorithms for Reg. Sets.

We consider a few important decision problems for reg. sets:

- The emptiness problem
- The finiteness problem
- The equivalence problem

The emptiness problem

Input. A DFA (NFA) M

Question. Is $L(M) = \phi$?

The finiteness problem is def. similarly.

The equivalence problem

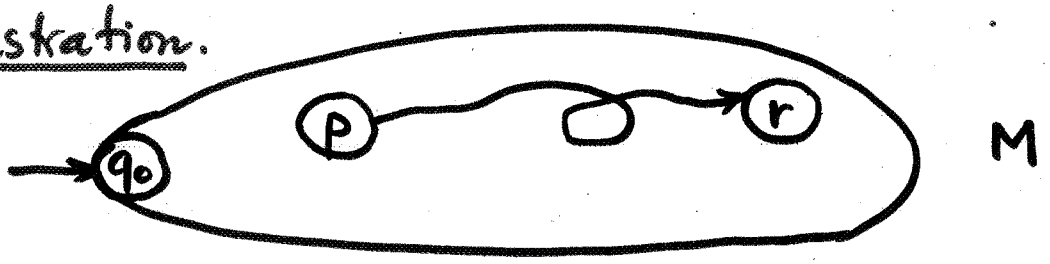
Input. DFAs M_1 and M_2

Question. Is $L(M_1) = L(M_2)$?

To solve the emptiness/finiteness problem it's useful to introduce the following.

Def. Let $M = (Q, \Sigma, \delta, q_0, F)$ be an FA (DFA or NFA), and $p, r \in Q$. Then r is **accessible** (**reachable**) from p if there is a path from p to r in M

Illustration.



For $p \in Q$:

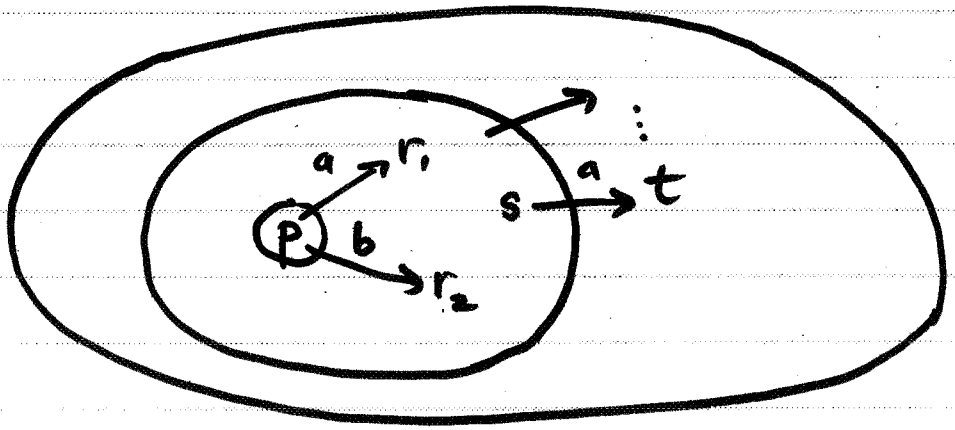
$Acc(p) :=$ set of states accessible from p (by path of length ≥ 0 incl. the empty path)

$Acc_+(p) :=$ set of states accessible from p by a nonempty path.

Q. Given M and p, can we efficiently compute $Acc(p)$ and $Acc_+(p)$?
(Note that $Acc(p) = Acc_+(p) \cup \{p\}$.)

Algorithm. (to compute $Acc_+(p)$)

Idea: Try to compute $Acc_+(p)$ inductively. First collect states accessible in one step from p. Then recursively collect states accessible from newly added states:



Stop when no new states can be added.

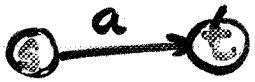
Input. FA $M = (Q, \Sigma, \delta, q_0, F)$, $p \in Q$

Output. $Acc_+(p)$

Method.

$A_1 := \{ r \in Q \mid \exists a \in \Sigma_\varepsilon : r \in \delta(p,a) \}$;

$i := 1$;



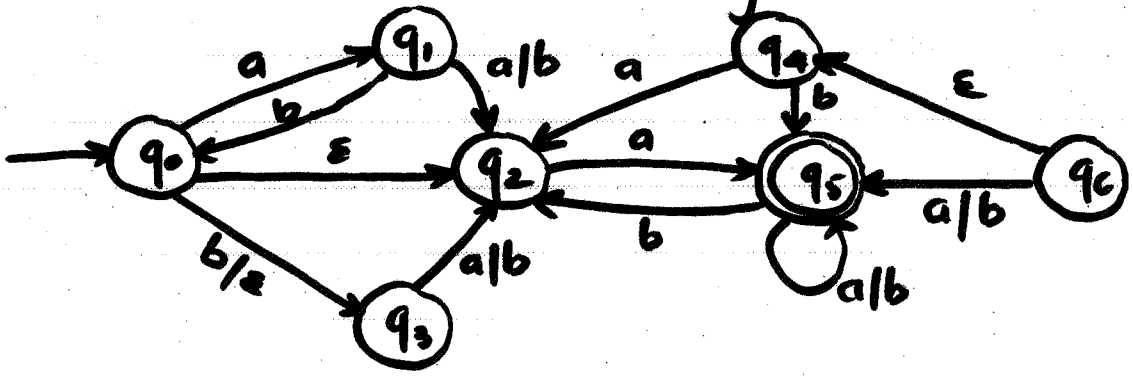
repeat $i := i+1$;

$A_i := A_{i-1} \cup \{ t \in Q \mid \exists s \in A_{i-1}, \exists a \in \Sigma_\varepsilon : t \in \delta(s,a) \}$;

until $A_i = A_{i-1}$;

$Acc_+(p) := A_i$

Ex. Consider the following NFA M:



We want to compute $Acc_+(q_0)$

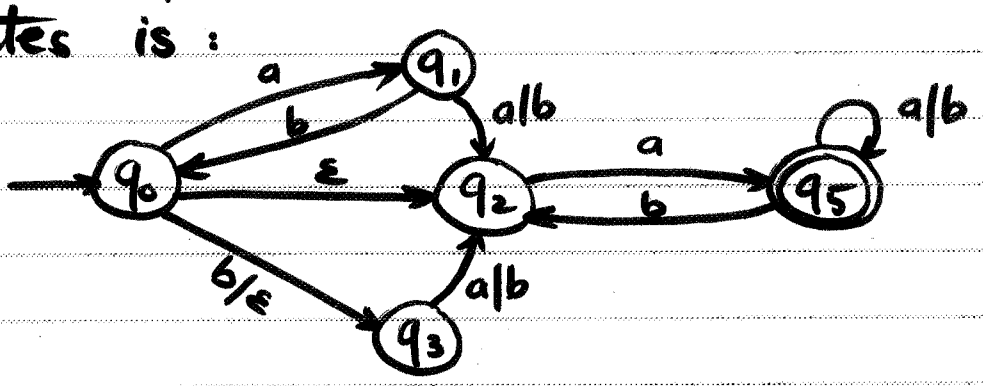
$A_1 = \{ q_1, q_2, q_3 \}$

$A_2 = \{ q_1, q_2, q_3, q_0, q_5 \}$

$A_3 = \{ q_1, q_2, q_3, q_0, q_5 \} = Acc_+(q_0)$

Note that q_4, q_6 are inaccessible from q_0: they can be removed without affecting $L(M)$.

The equiv. NFA without inaccess. states is:



□

Proposition. Every FA can be converted to an equiv. FA in which all states are accessible from initial state □

Proposition. The emptiness problem for reg. sets is solvable. That is there is an algorithm that decides for a given FA M whether $L(M) = \phi$.

Pf. Let $M = (Q, \Sigma, \delta, q_0, F)$ be given. Then

$$L(M) \neq \phi \iff Acc(q_0) \cap F \neq \phi.$$

Thus, given M we compute $Acc(q_0)$ and check if $Acc(q_0) \cap F \neq \phi$ □

Proposition. The finiteness problem for regular sets is solvable.

Pf. Suppose we are given a DFA

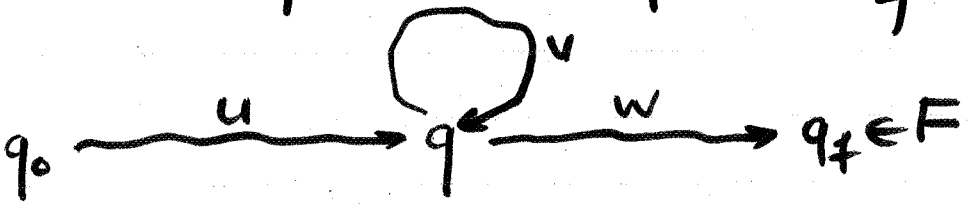
$$M = (Q, \Sigma, \delta, q_0, F)$$

Goal: To check whether $L(M)$ is finite or infinite.

Suppose that $L(M)$ is infinite. Then there exist arbitrarily long strings in $L(M)$.

Let $z \in L(M)$ and $|z| \geq n$, where $n = \text{card}(Q)$.

By P.L., z can be written as $z = uvw$ where v is label of a cycle around a state q on the computation path of z:



Thus,

$L(M)$ infinite $\Rightarrow \exists\, q \in Q$:

(1) $q \in \text{Acc}(q_0)$
(2) $q \in \text{Acc}_+(q)$
(3) $\text{Acc}(q) \cap F \neq \emptyset$

Since the converse of this also holds true, we have the following characterization:

$L(M)$ infinite $\iff \exists\, q \in Q$:

(1) $q \in Acc(q_0)$

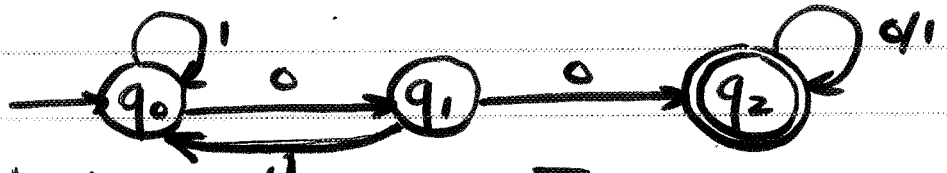
(2) $q \in Acc_+(q)$

(3) $Acc(q) \cap F \neq \emptyset$.

Thus, to check whether $L(M)$ is infinite, compute $Acc(q_0)$ and find a state q, compute $Acc(q)$, $Acc_+(q)$, and check (1), (2) & (3).

If some q can be found, then $L(M)$ is infinite; otherwise it's finite. □

Ex: Consider the DFA



Let q be q_2. Then:

(1) $q_2 \in Acc(q_0)$

(2) $q_2 \in Acc_+(q_2)$

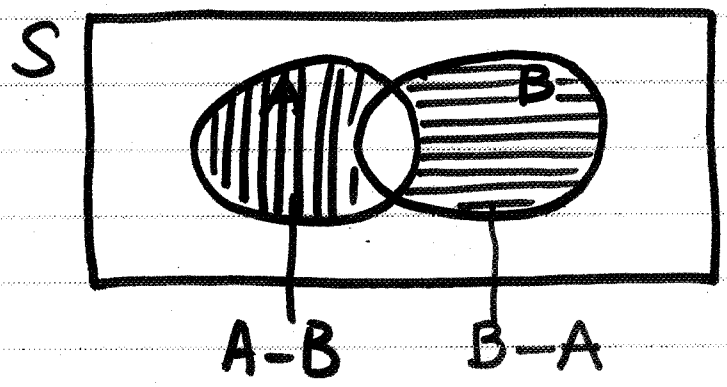
(3) $Acc(q_2) = \{q_2\} \cap F \neq \emptyset$ □

Proposition. The equivalence problem for regular sets is solvable.

Pf. Let M_1, M_2 be two DFAs.

Note that for sets $A, B \subseteq S$:

$$A = B \iff A - B = \emptyset \wedge B - A = \emptyset$$
$$\iff A \cap \overline{B} = \emptyset \wedge B \cap \overline{A} = \emptyset$$



Hence, $L(M_1) = L(M_2) \iff$

$$\underbrace{L(M_1) \cap \overline{L(M_2)}}_{(*)} = \emptyset \wedge \underbrace{L(M_2) \cap \overline{L(M_1)}}_{(**)} = \emptyset$$

From the proofs that reg. sets are closed under $\overline{\ }$ and $\cap$, we can construct DFAs M^* and M^{**} for (*) and (**) respectively. Next using the algor. for checking emptiness we can verify whether $L(M^*) = \emptyset$ and $L(M^{**}) = \emptyset$ $\square$

Applications.

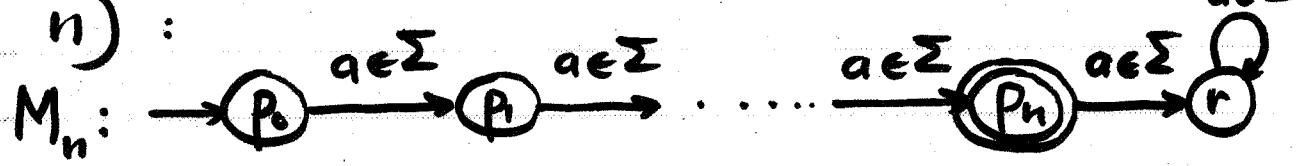
Consider the following decision probl.

Input. A DFA M and integer $n \geq 0$

Question. Does M accept some string of length n ?

Claim. The above decision problem is solvable.

Pf. Let $M = (Q, \Sigma, \delta, q_0, F)$ and n be given. We constr. a DFA M_n to accept strings in Σ^n (= set of strings of length n) :



Now construct DFA $N = M \times M_n$ to accept $L(M) \cap L(M_n)$.

Clearly N accepts strings of length n in L(M).

Now simply check if $L(N) = \emptyset$.

If $L(N) = \emptyset$, then M accepts no strings of length n; otherwise it does accept some string of length n □

Q. Can you modify the algor. to compute Acc(q) to obtain an algor. for the above probl ?
Consider another decision problem

Input. A DFA $M=(Q,\Sigma,\delta,q_0,F)$ and a string $w \in \Sigma^*$.

Question. Does M accept a string that contains the pattern w ?

Claim. This decision problem is solvable.

Pf. Let $M=(Q,\Sigma,\delta,q_0,F)$ be a DFA and $w \in \Sigma^*$.

Construct a DFA M_w that accepts $\Sigma^* w \Sigma^*$, the set of strings containing w as substring.

Now construct a DFA $N = M \times M_w$ that accepts $L(M) \cap L(M_w)$. Then

M accepts a string containing w as substring $\iff$ $L(M) \cap L(M_w) \neq \emptyset$

$\iff$ $L(N) \neq \emptyset$

Thus we simply check whether $L(N)$ is empty. □