3. Regular Languages

The family of regular languages
- *all finite language*
- *closed under union, concatenation and closure.*

Three equivalent ***descriptions*** *on relular languages*
- *regular expression*
- *finite automata*
- *regular grammar*

Section 3.1
- *regular expression*
- *relular language*

Section 3.2
- *finite automata*
- *regular expression* $\leftrightarrow$ *finite automata*

Section 3.3
- *regular grammar*
- *finite automata* $\leftrightarrow$ *regular grammar*

Section 3.4
- *deterministic finite automata* $\subseteq$ *finite automata*
- *finite automata* $\rightarrow$ *deterministic finite automata*

Section 3.5
- *some decision problems on regular languages*

Section 3.6
- *lexical analysis*

3.1 Regular Expression

Language description

any finite *means of specifying languages*

finite number of finitely structured elements

writing down all the sentences ***finite language***

rewriting system ***infinite language***

$L \subseteq V^*$ *languages over V*

countably infinite $\{0, 1\}^*$

$2^L \subseteq 2^{V^*}$ *family of languages over V*

uncountably infinite $\{0, 1\}^{\mathbb{N}}$

no description in general

regular expression over *an alphabet V*

well-formed expression

arguments $\in V \cup \{\underline{\varepsilon}, \underline{\emptyset}\}$

operators * *closure*

$\cdot$ *concatenation*

$\cup$ *union*

Let E be a string over $V \cup \{\underline{\varepsilon}, \underline{\varnothing}, {}^*, \cdot, \cup,), (\}$

(1) E is a ***regular primary over*** *V, if*

symbol in $V \cup \{\underline{\varepsilon}, \underline{\varnothing}\}$*; or*

(E_1) E_1 *is a regular expression over V.*

(2) E is a ***regular factor over*** *V, if*

regular primary over V; or

E_1^* E_1 *is a regular factor over V.*

(3) E is a ***regular term over*** *V, if*

regular factor over V; or

$E_1 \cdot E_2$ E_1 *is a regular term and* E_2 *is a regular factors over V.*

(4) E is a ***regular expression over*** *V, if*

regular term over V; or

$E_1 \cup E_2$ E_1 *is a regular expression and* E_2 *is a regular term over V.*

$$\langle P\rangle \to \underline{\varnothing} \mid \underline{\varepsilon} \mid \langle V\rangle \mid (\ \langle E\rangle\ )$$
$$\langle F\rangle \to \langle P\rangle \mid \langle F\rangle^*$$
$$\langle T\rangle \to \langle F\rangle \mid \langle T\rangle \cdot \langle F\rangle$$
$$\langle E\rangle \to \langle T\rangle \mid \langle E\rangle \cup \langle T\rangle$$

$$\langle E\rangle \to \underline{\varnothing} \mid \underline{\varepsilon} \mid \langle V\rangle \mid (\ \langle E\rangle\ ) \mid \langle E\rangle \cup \langle E\rangle \mid \langle E\rangle \cdot \langle E\rangle \mid \langle E\rangle^*$$

L(E) ***language denoted by*** *a regular expression E.*

(1a) $L(\underline{\emptyset}) = \emptyset$.

(1b) $L(\underline{\varepsilon}) = \{\varepsilon\}$.

(1c) $L(a) = \{a\}\ \forall a \in V$.

(1d) $L((E)) = L(E)\ \forall$ regular expression E over V

(2) $L(E^*) = (L(E))^*\ \forall$ regular factor E over V

(3) $L(E_1 \cdot E_2) = L(E_1) \cdot L(E_2)\ \forall$ r.t. E_1 and r.f. E_2 over V

(4) $L(E_1 \cup E_2) = L(E_1) \cup L(E_2)\ \forall$ r.e. E_1 and r.t. E_2 V

regular language *L over V*

L = L(E) for some ***regular expression*** *E over V*

Fact 3.2 *Any <u>finite</u> language is* ***regular****.*

A language family $\mathbb{F}$ *is* ***effectively closed*** *under n-ary operation f, if any n-tuple* $D_1, \dots, D_n$ *of language* ***descriptions****, can be* ***transformed*** *into a* ***description*** *of the language* $f(L(D_1), \dots, L(D_n))$.

A language family $\mathbb{F}$ *is* ***closed*** *under f if*

$L_1, \dots, L_n \in \mathbb{F}$ implies $f(L_1, \dots, L_n) \in \mathbb{F}$.

closed vs effectively closed

Lemma 3.3 *Let E be a regular expression over V. E can be transformed in time $O(|E|)$ into a regular expression that denotes $L(E)^*$. Any pair of regular expressions E_1 and E_2 can be transformed in time $O(|E_1| + |E_2|)$ into regular expressions that denotes $L(E_1)L(E_2)$ and $L(E_1) \cup L(E_2)$.*

$(E)^*, (E_1)(E_2), (E_1) \cup (E_2).$

Theorem 3.4 (Kleene) *For any alphabet V, the* ***family of regular languages*** *over V is the* ***smallest*** *family of languages over V that contains all finite language over V and is (**effectively**)* ***closed*** *under <u>closure</u>, <u>concatenation</u>, and <u>finite union</u>.*

Two language descriptions D_1 and D_2 are ***equivalent****, if $L(D_1) = L(D_2)$;* ***inequivalent,*** *otherwise.*

Fact 3.5 *For any regular expression there exists a* ***countably infinite*** *number of <u>equivalent</u> regular expressions.*
Proof*. $E, E \cup E, E \cup E \cup E, \ldots$.*

Given any class of language description $\mathbb{D}$, any description in $\mathbb{D}$ usually has a ***countably infinite*** *number of equivalent descriptions.*
(renaming of symbols)

*A language description D is **ambiguous** if some sentence in L(D) is **described in <u>two</u> ways**. A description D is **unambiguous** if it is not ambiguous.*

*(1) $\emptyset$, ε, a and (E) are **unambiguous**, if*
$^{\forall}a \in V$, E is <u>unambiguous</u>, respectively.
*(2) E^* is **unambiguous** if E is <u>unambiguou</u>s and*
$^{\forall}x \in L(E^*)$ $^{\exists}$one $n \geq 0$, $^{\exists}$one sequence $(x_1, \ldots, x_n)$
.э. $x = x_1 \ldots x_n$, $1 \leq i \leq n$, $x_i \in L(E)$.
*(3) E_1E_2 is **unambiguous**, if*
(a) $L(E_1E_2) = \emptyset$; or
(b) if E_1, E_2 are <u>unambiguous</u>, and
$^{\forall}x \in L(E_1E_2)$, $^{\exists}$one $(y, z) \in L(E_1) \times L(E_2)$
.э. $yz = x$.
*(3) $E_1 \cup E_2$ is **unambiguous,** if*
E_1, E_2 are <u>unambiguous</u>, $L(E_1) \cap L(E_2) = \emptyset$.

***ambiguity** in language description*
*it may have two different **meanings***
enhanced description power
$(0\cup 1)^*(000\cup 111)(0\cup 1)^*$

unambiguity of language description
may reduce the descriptive power

description power of unambiguous regular language

same as unrestricted one

No "inherently ambiguous" *regular languages*

Theorem 3.6 Any regular expression can be transformed into unambiguous one.

Proof

regular expression $\Rightarrow$ *deterministic automaton*

unambiguous automaton $\Rightarrow$ *unambiguous r.e.*

deterministic automaton $\subseteq$ *unambiguous automaton*

The descriptions in $\mathbb{D}_1$ *are* ***at least as descriptive as*** *those in* $\mathbb{D}_2$*, if*

$$^{\forall}D_2 \in \mathbb{D}_2, {}^{\exists}D_1 \in \mathbb{D}_1, L(D_1) = L(D_2).$$

$$(L(\mathbb{D}_1) \supseteq L(\mathbb{D}_2))$$

The description in $\mathbb{D}_1$ *are* ***more descriptive as*** *those in* $\mathbb{D}_2$*, if the descriptions in* $\mathbb{D}_1$ *are at least as descriptive as those in* $\mathbb{D}_2$ *but*

$$^{\exists}D_1 \in \mathbb{D}_1, {}^{\nexists}D_2 \in \mathbb{D}_2, L(D_1) = L(D_2).$$

$$(L(\mathbb{D}_1) \supset L(\mathbb{D}_2))$$

The descriptions in $\mathbb{D}_1$ *are* ***as descriptive as*** *those in* $\mathbb{D}_2$*, if the descriptions in* $\mathbb{D}_1$ *are at least as descriptive as those* $\mathbb{D}_2$ *and vice versa.*

$$(L(\mathbb{D}_1) = L(\mathbb{D}_2)).$$

The descriptions in $\mathbb{D}_1$ *are* ***at least as succinct as*** *those in* $\mathbb{D}_2$, *if*

$\forall D_2 \in \mathbb{D}_2, \exists D_1 \in \mathbb{D}_1, L(D_1) = L(D_2),$

the size of D_1 *is at most linear in the size of* D_2.

The descriptions in $\mathbb{D}_1$ *are* ***as succinct as****(or* ***equivalent in succinctness****) those in* $\mathbb{D}_2$, *if …*

Let f be a function: natural numbers $\to$ *positive reals*

The descriptions in $\mathbb{D}_1$ *can be* ***f(n) more succinct than*** *those in* $\mathbb{D}_2$, *if* $\exists L_1, \ldots L_n, \ldots$ *each* L_n *has description in* $\mathbb{D}_1$, *size of* L_n *is O(n) but the equivalent description in* $\mathbb{D}_2$ *are of size at least f(n).*

Proposition 3.7 *There exist a constant* $c > 0$ *and an infinite sequence of regular languages* $L_1, L_2, \ldots$ *over* $\{0, 1\}$ *such that each* L_n *is denoted by an* ***ambiguous regular expression*** *of length O(n) but any* ***unambiguous regular expression*** *denoting* L_n *must have length at least* $2^{c \cdot n}$.

(Ambiguous) *regular expressions are*

2^n ***more succinct than***

unambiguous *regular expressions.*

3.2 Finite Automata

Let $M = (V, P)$ be a rewriting system.

$Q \cup \Sigma = V$, $Q \cap \Sigma = \emptyset$, $q_s \in Q$, *and* $F \subseteq Q$.

$M = (Q, \Sigma, P, q_s, F)$ *is a* ***finite automaton*** *with*

state alphabet	Q,
input alphabet	Σ,
initial state	q_s,
set of final states	F,
rules	P, *if*
$qx \to p$	$q, p \in Q, x \in \Sigma^*$.

x ***transition***(***transition*** *on* x) *from state* q *to* p.

configuration(***instantaneous description***) *of* M

qw $\quad$ $qw \in Q\Sigma^*$.

qw $\quad$ ***initial*** *for* $w \in \Sigma^*$, *if* $q = q_s$.

qw $\quad$ ***accepting***, *if* $w = \varepsilon$, $q \in F$.

qw $\quad$ ***error***, *if nonaccepting,*

$0 \le {}^\forall k \le |w|$, $qk{:}w \to p \notin P$.

computation(***process***) *of* M *on input string* w

derivation in M from <u>initial</u> <u>configuration</u> $q_s w$

accepting *computation,*

if it end with <u>accepting</u> configuration

M ***accepts*** w, *if it has an accepting computation.*

***language accepted*(*recognized* or *described*) by M**

$L(M) = \{w \in \Sigma^* \mid q_s w \overset{*}{\Rightarrow} q \text{ in } M, q \in F\}$.

transition graph

nodes Q

edges (q, p) $qx \rightarrow p \in P$

labeled by x

A finite automaton is ***ambiguous****,*

if it accepts some sentence in two distinct ways

$\exists$*at least two accepting computation,*

it is ***unambiguous****, otherwise.*

A state p is ***reachable*** *from a state q upon* ***reading*** *w*

$qw \overset{*}{\Rightarrow} p$.

A state p is ***accessible upon reading*** *w,*

$q_s w \overset{*}{\Rightarrow} p$.

Fact 3.8 *A finite automaton accepts w, iff some final state is accessible upon reading w.*

accessible *state*

A state that is accessible upon reading some string

inaccessible *state, if it is not accessible.*

A fa with no inaccessible state is called ***reduced.***

Lemma 3.9 *The set of states* ***reachable*** *from a given state of a finite automaton M can be computed in time O(|M|).*

Proof.

q_1 ***reaches*** q_2, *if M has a transition from* q_1 *to* q_2, *in O(|M|)*

q_1 ***reaches**** q_2, *in O(|M|) by theorem 2.2*

Theorem 3.10 *Any finite automaton* $M(Q, \Sigma, P, q_s, F)$ *can be transformed in time O(|M|) into an* ***equivalent reduced*** *finite automaton* $M(Q', \Sigma, P', q_s, F')$, *where* $Q' \subseteq Q$, $P' \subseteq P$, *and* $F' \subseteq F$.

A state q is ***live****, it some final state is reachable from it. A state that is not live is* ***dead****.*

Lemma 3.11 *The set of* **live** *states of a finite automaton M can be computed in time O(|M|).*

Proof. *(set of live states)* ($\textbf{\textit{reaches}}^{-1})^*$ *F.*

Fact 3.12 *Let M be an* ***unambiguous*** *finite automaton, Then for any accessible live state* q_1 *and* q_2, *and any input string x, there is* ***at most one derivation*** *of* q_2 *from* q_1x *in M.*

A finite automaton is ***normal-form****, if*

$q_1 t \to q_2 \in P,\ t \in \Sigma \cup \{\varepsilon\}$.

A finite automaton is ***ε-free****, if no ε-transition.*

Normal-form *finite automaton are equivalent in* <u>*description power*</u> *as well as* ***succinctness*** *to unrestricted finite automaton.*

Theorem 3.13 *Any fa* $M = (Q, \Sigma, P, q_s, F)$ *can be transformed in time* $O(|M|)$ *into an equivalent* ***normal-form*** *fa* $M' = (Q', \Sigma, P', q_s', F')$. *Moreover M' is* ***ambiguous*** *iff M is; and M' is* ***ε-free*** *iff M is.*

Proof*.*

$Q' = \{[q] \mid q \in Q\} \cup \{[qx] \mid qxy \to p \in P, y \in \Sigma^+\}$

$q_s' = [q_s]$,

$F' = \{[q] \mid q \in F\}$.

$P' = \{[q] \to [p] \mid q \to p \in P\}$

$\cup \{[qx]a \to [qxa] \mid a \in \Sigma, qxay \to p \in P, y \in \Sigma^+\}$

$\cup \{[qx]a \to [p] \mid a \in \Sigma, qxa \to p \in P\}$

i) $q \to p \in P$

$[qw] \Rightarrow_{M'} [pw]$.

ii) $qa_1 \ldots a_n \to p \in P, n \geq 1$

$[q]a_1 \ldots a_n w \Rightarrow_{M'} [qa_1]a_2 \ldots a_n w \Rightarrow_{M'} \ldots \Rightarrow_{M'} [qa_1 \ldots a_{n-1}]a_n w \Rightarrow_{M'} [p]w$ in M'

$\therefore qx \to p \in P$, iff $[q]xw \Rightarrow^* [p]w$ in M'.

If $qa_1 \ldots a_n \to p \in P, n \geq 1$

$[q]a_1 \to [qa_1]$,

$[qa_1]a_2 \to [qa_1a_2]$,

$\ldots$,

$[qa_1 \ldots a_{n-2}]a_{n-1} \to [qa_1 \ldots a_{n-1}]$,

$[qa_1 \ldots a_{n-1}]a_n \to p \in P'$

$|P|: (n+2) \Rightarrow |P'|: 3n$

Lemma 3.14 *Any* <u>*normal-form*</u> *fa* $M = (Q, \Sigma, P, q_s, F)$ *can be transformed in time* $O(|Q| \cdot |M|)$ *into an equivalent* ε***-free*** *fa* $M'' = (Q, \Sigma, P'', q_s, F'')$. *Moreover if M is unambiguous, then so is M''.*

Proof

$P'' = \{qa \to p \mid \exists q'' \ni q \stackrel{*}{\Rightarrow} q'', q''a \to p \in P\}$.

$F'' = \{q \mid q \stackrel{*}{\Rightarrow} q'' \in F\}$

$\therefore$ $qw \stackrel{*}{\Rightarrow} p$ *in M, iff* $qw \stackrel{*}{\Rightarrow} q''$ *and* $q'' \stackrel{*}{\Rightarrow} p$ *in M.*

M'' may remove some ambiguities on sequences of ε*-moves.*

q ***empty-trans*** *p, iff* $q \to p \in P$ *of size* $O(|M|)$.

q ***empty-trans****** *p, iff* $q \stackrel{*}{\Rightarrow} p \in P$

in time $O(|Q| \cdot |M|)$ *Theorem 2.3.*

Theorem 3.15 *Any fa M can be transformed in time $O(|M|^2)$ into an equivalent ε-**free normal-form fa**. Moreover if M is unambiguous, so is transformed automaton.*

*Nonlinear time bound in **theorem 3.15***

*fa with ε-transition is $O(|M|^2)$ more **succinct** than*

ε-free counterpart

L_1, L_2, … each L_n non-ε-free n.f. fa O(n)

*ε-**free normal-form fa** at least 3n(n+1)/2.*

finite automaton

⇓ *O(n) in succinctness*

reduced fa

⇓ *O(n) in succinctness*

normal form fa

⇓ *$O(n^2)$ in succinctness*

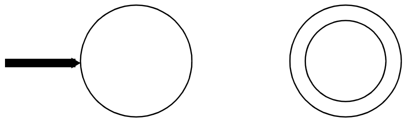
ε-free fa

e-free normal-form fa at least 3n(n+1)/2.

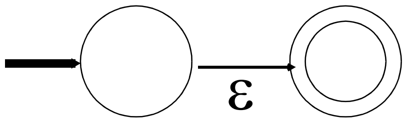
Exercises 3.7

Theorem 3.16 *Any* ***regular expression*** *E over Σ can be transformed in time O(|E|) into an* ***equivalent finite automaton*** *M(E) with input alphabet Σ. Moreover M(E) is unambiguous iff E is.*

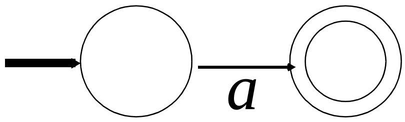
(1a) M(∅)



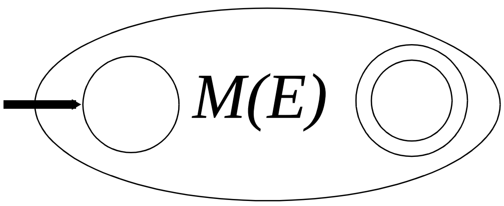
(1b) M(ε)



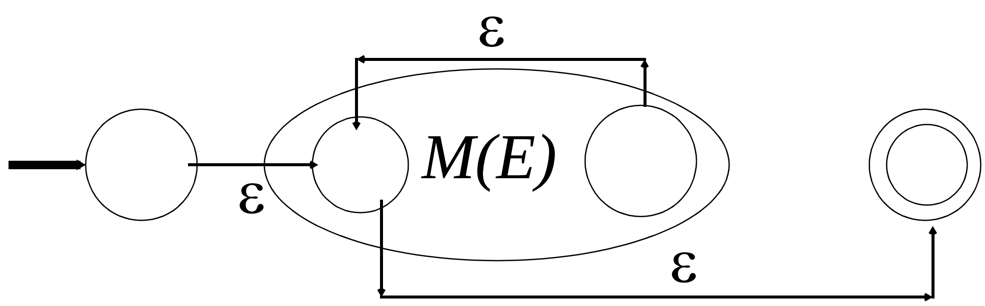
(1c) M(a)



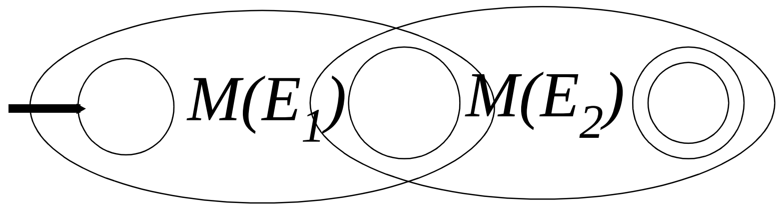
(1d) M((E))



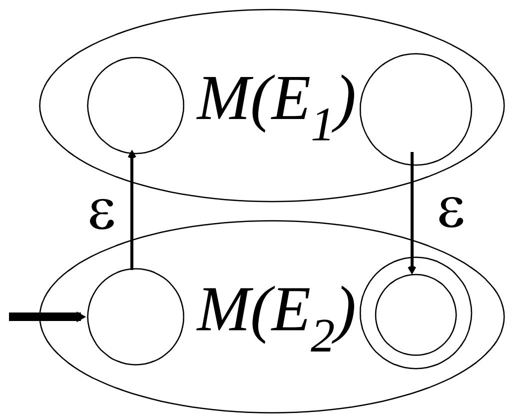
(2) $M(E^*)$



(3) $M(E_1E_2)$



(4) $M(E_1 \cup E_2)$



$L_{match} = \{0^n 1^n \mid n \geq 0\}$ is not regular.

Proof ***pumping***

Assume L_{match} is regular.

ε-free and normal-form automaton M.

assume $n = |Q| + 1$

$$q_s 0^n 1^n \overset{n}{\Rightarrow} q 1^n \overset{n}{\Rightarrow} q_f \in F$$

Since $n > |Q|$, $^\exists i \geq 0$, $k > 0$, and $^\exists p \in Q$.

$$q_s 0^n 1^n \overset{i}{\Rightarrow} \mathbf{p} 0^{n-i} 1^n \overset{k}{\Rightarrow} \mathbf{p} 0^{n-i-k} 1^n \overset{n-i-k}{\Rightarrow} q 1^n \overset{n}{\Rightarrow} q_f \in F..$$

$\therefore q_s 0^i 0^{j\cdot k} 0^{n-i-k} 1^n \; (= 0^{n-k+j\cdot k}) \overset{*}{\Rightarrow} q_f \in F, \; ^\forall j \geq 0.$

but $0^i 0^{j\cdot k} 0^{n-i-k} 1^n \notin L_{match}$.

Theorem 3.17 *Any fa $M = (Q, \Sigma, P, q_s, F)$ can be transformed in time $O(|Q|\cdot|M|\cdot 4^{|Q|})$ into an equivalent* ***regular expression*** *$E(M)$ over Σ. Moreover M is* ***ambiguous*** *iff M is.*

Proof *Let $Q = \{q_1, \ldots, q_n\}$, for $1 \leq i, j \leq n$, $0 \leq k \leq n$*

E_{ij}^k a regular expression, $x \in L(E_{ij}^k)$

q_j is reachable$^{(\text{one or more})}$ from q_i upon reading x

without going through any state q_m .э. $m > k$.

$$L(E_{ij}^k) = \{x_0 \in \Sigma^* \mid q_{s_0} x_0 \Rightarrow q_{s_1} x_1 \ldots \Rightarrow q_{s_m} x_m,\ m \geq 1,$$

$$q_{s_0} = q_i,\ q_{s_m} = q_j,\ x_m = \varepsilon,\ s_l \leq k,\ 1 \leq {}^\forall l \leq m-1\}.$$

For $k = 0$, $E_{ij}^{0} = x_1 \cup \ldots \cup x_m$ where

$$q_i x_l \rightarrow q_j \in P, \text{ for } 1 \leq l \leq m,$$

$$= \varnothing, \text{ otherwise.}$$

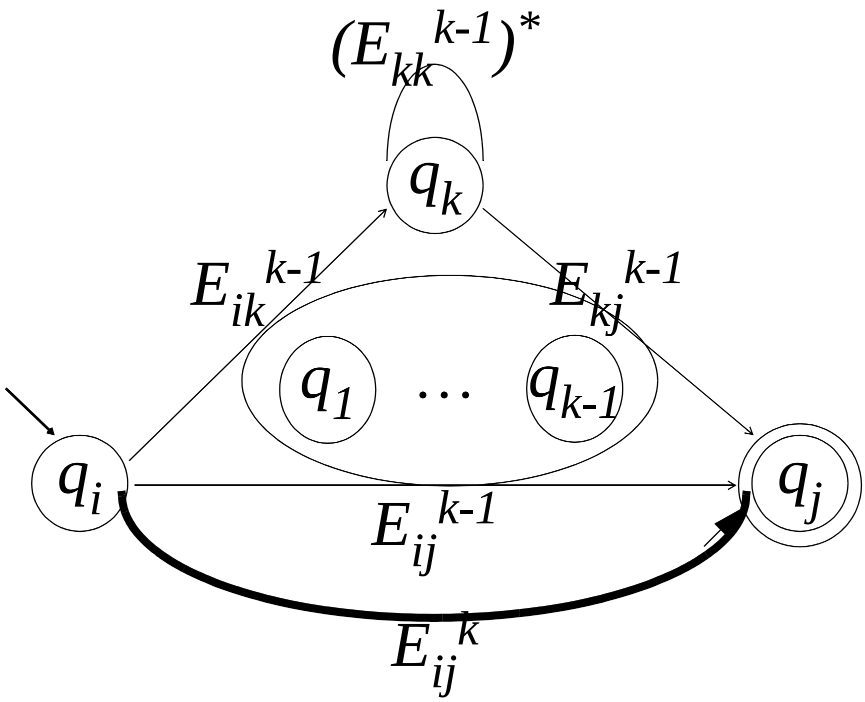
For $k > 0$,

(1) $E_{ij}^{k} = (E_{ij}^{k-1})^{+}$, $\quad i=j=k$.

(2) $E_{ij}^{k} = E_{ij}^{k-1} \cdot (E_{jk}^{k-1})^{*}$ $\quad i \neq j=k$.

(3) $E_{ij}^{k} = (E_{ik}^{k-1})^{*} \cdot E_{kj}^{k-1}$ $\quad i=k \neq j$.

(4) $E_{ij}^{k} = E_{ij}^{k-1} \cup E_{ik}^{k-1} \cdot (E_{kk}^{k-1})^{*} \cdot E_{kj}^{k-1}$ $\quad i \neq k \neq j$.



$E(M) = E_{sf_1}^{\,n} \cup \ldots \cup E_{sf_m}^{\,n} \cup E_{\varepsilon}$.

where q_s is a initial state, $\{q_{f_1}, \ldots, q_{f_m}\} = F$, and

$E_{\varepsilon} = \varepsilon$, if $q_s \in F$; $E_{\varepsilon} = \varnothing$, $q_s \notin F$.

Theorem 3.18 *A language over Σ is **regular**, if and only if it is the language accepted by some **automaton** with input alphabet Σ.*

fa ⇒ re exponential time bound

re ⇒ fa linear time bound

Infinite sequence of regular languages $L_1, L_2, \dots$

L_n *ε-free normal form fa* $O(n^2)$*,*

regular expression 2^n*.*

*fa can be **exponentially more succinct** than re.*

finite automaton

⇓ *O(n) in succinctness*

reduced fa

⇓ *O(n) in succinctness*

normal form fa

⇓ $O(n^2)$ *in succinctness*

ε-free fa

⇓ $O(2^n)$ *in succinctness*

regular expression

3.3 Regular Grammars

Let G = (V, P) be a rewriting system.

$N \cup \Sigma = V$, $N \cap \Sigma = \emptyset$, and $S \in N$.

G = (N, Σ, P, S) is a ***right linear grammar*** *with*

nonterminal alphabet	*N,*
terminal alphabet	*Σ,*
start symbol	*S, and*
rules	*P, if*

$$A \rightarrow x, \quad A \rightarrow xB, \quad A,B \in N, x \in \Sigma^*.$$

G = (N, Σ, P, S) is a ***left linear grammar****, if*

$$A \rightarrow x, \quad A \rightarrow Bx, \quad A,B \in N, x \in \Sigma^*.$$

A rewriting system is a ***regular grammar****, if it is either* <u>*right*</u> <u>*linear*</u> *or* <u>*left*</u> <u>*linear*</u>*.*

language generated(described) by *G*

$$L(G) = L_G(S) = \{w \in \Sigma^* \mid S \stackrel{*}{\Rightarrow} w \text{ in } G\}.$$

sentential forms *in* ***regular grammar***

$$S \stackrel{*}{\Rightarrow} xA \Rightarrow xyB \stackrel{*}{\Rightarrow} xyz, \; A \rightarrow yB \in P.$$

<u>*right*</u> *linear* ***single*** <u>*rightmost*</u> *nonterminal*

$$S \stackrel{*}{\Rightarrow} Ax \Rightarrow Byx \stackrel{*}{\Rightarrow} zyx, \; A \rightarrow By \in P.$$

<u>*left*</u> *leaner* ***single*** <u>*leftmost*</u> *nonterminal*

$$q_s xyz \stackrel{*}{\Rightarrow} q_A yz \Rightarrow q_B z \stackrel{*}{\Rightarrow} q_F, \; q_A y \rightarrow q_B \in P, \; q_F \in F.$$

G is ***ambiguous****, if some sentence in L(G) has* <u>*two distinct*</u> *derivations; otherwise* ***unambiguous.***

Theorem 3.19 Any fa $M = (Q, \Sigma, P_M, q_s, F)$ can be transformed in time $O(|M|)$ into an equivalent ***right-linear grammar*** *$G(M) = (N, \Sigma, P_G, S)$. Moreover $G(M)$ is unambiguous if and only if M is.*

Proof $N = Q$, $S = q_s$, and

$$P_G = \{p \to xq \mid px \to q \in P_M\} \cup \{f \to \varepsilon \mid f \in F\}.$$

Theorem 3.20 *Any r.l.g. $G = (N, \Sigma, P_G, S)$ can be transformed in time $O(|M|)$ into an equivalent* ***finite automaton*** *$M(G) = (Q, \Sigma, P_M, q_s, F)$. Moreover $M(G)$ is unambiguous if and only if G is.*

Proof $q_s = [S]$,

$$Q = \{[A] \mid A \in N\} \cup \{[xA] \mid A \to x \in P_G, x \neq \varepsilon\},$$

$$P_M = \{[A]x \to [B] \mid A \to xB \in P_G\}$$
$$\cup \{[A]x \to [xA] \mid A \to x \in P_G, x \neq \varepsilon\},$$

$$F = \{[A] \mid A \to \varepsilon \in P_G\} \cup \{[xA] \mid A \to x \in P_G, x \neq \varepsilon\}.$$

right linear grammar finite automaton

same ***succinctness*** *and* ***descriptive power***

Theorem 3.21 *Any language over Σ is* ***regular*** *if and only if the language is generated by some* ***right linear grammar*** *over Σ.*

Theorem 3.23 *Any* <u>*right-linear grammar*</u> $G_r = (N_r, \Sigma, P_r, S_r)$ *can be transformed in time* $O(|M|)$ *into an equivalent* ***left-linear grammar*** $G_l = (N_l, \Sigma, P_l, S_l)$. *Moreover* G_l *is unambiguous if and only if* G_r *is.*

Proof. $N_l = N \cup \{S_l\}$

$P_l = \{B \to Ax \mid A \to xB \in P_r\}$

$\quad \cup \{S_l \to Ax \mid A \to x \in P_r\} \cup \{S_r \to \varepsilon\}$

$S_r = A_0 \Rightarrow x_1A_1 \Rightarrow \ldots \Rightarrow x_1\ldots x_{n-1}A_{n-1} \Rightarrow x_1\ldots x_n$ in G_r.

$\quad A_{i-1} \to x_iA_i \in P_r\ 1 \le i < n,\ A_{n-1} \to x_n \in P_r.$

$$\Downarrow\Uparrow$$

$\quad S_l \to A_{n-1}x_n \in P_l,\ A_i \to A_{i-1}x_i \in P_l\ 1 \le i < n,\ S_r \to \varepsilon \in P_l.$

$S_l \Rightarrow A_{n-1}x_n \Rightarrow A_{n-2}x_{n-1}x_n \Rightarrow \ldots \Rightarrow A_1x_2\ldots x_n \Rightarrow A_0x_1\ldots x_n$
$= S_rx_1\ldots x_n \Rightarrow x_1\ldots x_n$ in G_l.

$S_r = A_0$ $\qquad\qquad\qquad\qquad$ S_l

x_1A_1 $\qquad\qquad\qquad\qquad$ $A_{n-1}x_n$

x_2A_2 $\qquad\qquad\qquad\qquad$ $A_{n-2}x_{n-1}$

$x_{n-1}A_{n-1}$ $\qquad\qquad$ A_1x_2

$A_0 = S_rx_1$

x_n $\qquad$ ε

Theorem 3.22 *Any* <u>*left-linear grammar*</u> $G_l = (N_l, \Sigma, P_l, S_l)$ *can be transformed in time* $O(|M|)$ *into an equivalent* ***right-linear grammar*** $G_r = (N_r, \Sigma, P_r, S_r)$. *Moreover* G_r *is unambiguous iff* G_l *is.*

Proof. $N_r = N_l \cup \{S_r\}$

$$P_r = \{B \to xA \mid A \to Bx \in P_l\}$$
$$\cup \{S_r \to xA \mid A \to x \in P_l\} \cup \{S_l \to \varepsilon\}$$

Theorem 3.24 *Any language over an alphabet* Σ *is* ***regular*** *if and only if it is the language generated by some* ***regular grammar*** *with input string* Σ.

*The **reversal** of a rule* $r = \alpha \to \beta$, $r^R = \alpha^R \to \beta^R$.

*The **reversal** of G,* $G^R = (V, P^R)$.

$$P^R = \{r^R \mid r \in P\}$$

Lemma 3.25 *Let* $G = (V, P)$ *be a rewriting system,* $\pi = r_1 \ldots r_n \in P^*$, $\gamma_1, \gamma_2 \in V^*$. *Then*

$$\gamma_1 \overset{\pi}{\Rightarrow} \gamma_2 \text{ in } G, \text{ iff } \gamma_1^R \overset{\pi^R}{\Rightarrow} \gamma_2^R \text{ in } G^R,$$

where $\pi^R = r_1^R \ldots r_n^R$.

Proof.

"only if"

i) $n=0$, $\pi = \varepsilon$, *trivial.*

ii) $n>0$, *let* $\pi = \mu r_n$, $r_n = \omega_1 \to \omega_2$.

$$\gamma_1 \overset{\mu}{\Rightarrow} \alpha\omega_1\beta \overset{r_n}{\Rightarrow} \alpha\omega_2\beta \ (= \gamma_2) \text{ in } G.$$

$$\gamma_1^R \overset{\mu^R}{\Rightarrow} (\alpha\omega_1\beta)^R \ (=\beta^R\omega_1^R\alpha^R) \text{ in } G^R \text{ by IH.}$$

$$\therefore \quad \beta^R\omega_1^R\alpha^R \overset{r_n^R}{\Rightarrow} \beta^R\omega_2^R\alpha^R = (\alpha\omega_2\beta)^R = \gamma_2^R.$$

$$\gamma_1^R \overset{\mu^R r_n^R}{\Rightarrow} \gamma_2^R.$$

"if" trivial, since

$$\text{if } \gamma_1^R \overset{\pi^R}{\Rightarrow} \gamma_2^R \text{ in } G^R, (\gamma_1^R)^R = \gamma_1 \overset{\pi^{RR}}{\Rightarrow} = \overset{\pi}{\Rightarrow} (\gamma_2^R)^R = \gamma_2.$$

Theorem 3.26 *Any rg G can be transformed in time $O(|G|)$ into a* ***regular grammar*** *G^R such that*

$$L(G^R) = L(G)^R.$$

Proof

G^R is left(right)-linear if G is right(left)-linear.
$|G^R| = |G|$ and G^R can be constructed in time $O(|G|)$.

Theorem 3.27 *Family of regular grammar is* ***effectively closed*** *under* ***reversal****.*

3.4 Deterministic Automaton

*A finite automation M is **nondeterministic**, if*

$qw \Rightarrow^{r_1} q_1w_1$, $qw \Rightarrow^{r_2} q_2w_2$, and $r_1 \neq r_2$.

*M is **deterministic**, if not nondeterministic.*

Fact 3.28 *A fa is **nondeterministic**, if and only if*

$qx \to q_1$, $qy \to q_2$, *where y is a **prefix** of x.*

Fact 3.29 *A **deterministic** fa is **unambiguous**, provided it has no ε-transitions from final states.*

no different accepting states.

deterministic ⊂ unambiguous

Let $R \subseteq Q$. *Then* $\delta_x(R) = \{p \mid qx \to p \in P, q \in R\}$.

$\delta_x: 2^Q \times \Sigma^* \to 2^Q$.

Let $\delta_x(q) = \delta_x(\{q\}) = \{p \mid qx \to p \in P, q \in R\}$.

Let $\delta_x^n(R) = \{p \mid qx \overset{n}{\Rightarrow} p \text{ in } M, q \in R\}$.

Let $\delta_x^*(R) = \{p \mid qx \overset{*}{\Rightarrow} p \text{ in } M, q \in R\}$.

Theorem 3.30 *Any fa M can be transformed in time* $O(2^{|M|+\log|M|+\log|\Sigma|})$ *into an equivalent **deterministic ε-free normal-form** fa* $\hat{M}$ *of size* $O(2^{|M|+\log|\Sigma|})$.

Proof *Assume* $M = (Q, \Sigma, P, q_s, F)$ *is normal-form.*

$\hat{M} = (\hat{Q}, \Sigma, \hat{P}, \hat{q}_s, \hat{F})$ *where*

$\hat{Q} = 2^Q$,

$\hat{q}_s = \{q \in Q \mid q_s \stackrel{*}{\Rightarrow} q \text{ in } M\}$; or $\delta_\varepsilon^*(\{q_s\})$

$\hat{F} = \{\hat{q} \in \hat{Q} \mid \hat{q} \cap F \neq \emptyset\}$, and

$\hat{P} = \{\hat{q}_1 a \rightarrow \hat{q}_2 \mid \hat{q}_1 \subseteq Q, a \in \Sigma, \hat{q}_2 = \delta_a^*(\hat{q}_1)\}$

where

$\delta_a^*(\hat{q}_1) = \{q_2 \in Q \mid q_1 a \stackrel{*}{\Rightarrow} q_2 \text{ in } M, q_1 \in \hat{q}_1\} \in \hat{Q}$.

$\hat{q}_1 \stackrel{*}{\Rightarrow} w\hat{q}_2$ in $\hat{M}$, iff $\hat{q}_2 = \delta_w^*(\hat{q}_1)$ *where*

$\delta_w^*(\hat{q}_1) = \{q_2 \in Q \mid q_1 w \stackrel{*}{\Rightarrow} q_2 \text{ in } M, q_1 \in \hat{q}_1\}$.

$L(\hat{M}) = \{w \in \Sigma \mid \hat{q}_s w \stackrel{*}{\Rightarrow} \hat{q}, \hat{q} \in \hat{F}\}$

$= \{w \in \Sigma \mid \delta_w^*(\hat{q}_s) \cap F \neq \emptyset\}$

$= \{w \in \Sigma \mid q_s w \stackrel{*}{\Rightarrow} q, q \in F\}$.

$|\hat{M}| = 3 \cdot |\hat{P}| = 3 \cdot |\hat{Q}| \cdot |\Sigma| = 3 \cdot |2^{|Q|}| \cdot |\Sigma| = 3 \cdot 2^{|Q| + \log|\Sigma|}$.

$\therefore\ |\hat{M}| = O(2^{|Q| + \log|\Sigma|})$.

time complexity

$O(|\hat{M}| \cdot |Q|) = O(2^{|Q| + \log|Q| + \log|\Sigma|})$

$q_1 \overset{*}{\Rightarrow} q_2$, *if and only if* $q_1\ \delta^*_\varepsilon\ q_2$.

$q_1 \overset{*}{\Rightarrow} aq_2$, *if and only if* $q_1\ \delta^*_\varepsilon \delta_a \delta^*_\varepsilon\ q_2$

$$\delta^*_a = \delta^*_\varepsilon \delta_a \delta^*_\varepsilon.$$

Furthermore if $w = a_1 \ldots a_n$.

$$\begin{aligned} \delta^*_w &= \delta^*_\varepsilon \delta_{a_1} \delta^*_\varepsilon \delta^*_\varepsilon \delta_{a_2} \delta^*_\varepsilon \ldots \delta^*_\varepsilon \delta_{a_n} \delta^*_\varepsilon, \\ &= \delta^*_\varepsilon \delta_{a_1} \delta^*_\varepsilon \delta_{a_2} \ldots \delta^*_\varepsilon \delta_{a_n} \delta^*_\varepsilon. \end{aligned}$$

define $\hat{\delta}_a = \delta_a \delta^*_\varepsilon$.

$$\begin{aligned} \delta^*_w &= \delta^*_\varepsilon \hat{\delta}_{a_1} \hat{\delta}_{a_2} \ldots \hat{\delta}_{a_n} \\ &= \delta^*_\varepsilon \hat{\delta}_w. \end{aligned}$$

$\hat{q}_s := \delta^*_\varepsilon(q_s);$

$\hat{Q} := \{\hat{q}_s\};\ \hat{P} := \emptyset;$

repeat

    for $\hat{q}_1 \in \hat{Q}$ ***and*** $a \in \Sigma$ ***do***

        $\hat{Q} := \hat{Q} \cup\ \hat{\delta}_a(\hat{q}_1);\ (\hat{\delta}_a = \delta_a \delta^*_\varepsilon)$

        $\hat{P} := \hat{P} \cup\ \hat{q}_1 \cdot a \rightarrow \hat{\delta}_a(\hat{q}_1);$

    od

until no more rule is added into $\hat{P}$;

$\hat{F} := \{\hat{q} \in \hat{Q} \mid \hat{q} \cap F \neq \emptyset\}.$

Theorem 3.31
(Characterization of Regular Languages)
The following statements are logically equivalent for all languages over alphabet Σ.

(1) L is the language denoted by some ***regular expression*** *over Σ.*
(2) L is the language denoted by some ***unambiguous regular expression*** *over Σ.*
(3) L is the language accepted by some ***finite automaton*** *with input alphabet Σ.*
(4) L is the language accepted by some ***deterministic ε-free finite automaton*** *with ….*
(5) L is the language generated by some ***regular grammar*** *with terminal alphabet Σ.*
(6) L is the language generated by some ***unambiguous right-linear grammar*** *with ….*
(7) L is the language generated by some ***unambiguous left-linear grammar*** *with ….*

Moreover, if D is a description of L belonging to any of the above classes of regular language description, then D can be transformed into a equivalent description belonging to any of the other classes.

finite automaton $\Leftrightarrow$ *regular grammar*

O(n) in succinctness

finite automaton $\Rightarrow$ *regular expression*

$O(2^n)$ in succinctness

finite automaton(regular grammar)

O(n) in succinctness

reduced fa(reduced rg)

O(n) in succinctness

normal form fa(normal form rg)

$O(n^2)$ in succinctness

ε-free fa(ε-free rg)

$O(2^n)$ in succinctness

deterministic fa(deterministic rg)

$O(2^n)$ in succinctness

regular expression

$O(2^n)$ in succinctness

unambiguous(deterministic) re

$O(2^{2^n})$

3.5 Decision Problems on Regular Languages

Let $\mathbb{D}$ be a class of regular language description, L be a regular language.

$P_{mem}(\mathbb{D})$: "Given w, $D \in \mathbb{D}$; is $w \in L(D)$?"
***membership problem** for $\mathbb{D}$.* *rep(D)#w*

$P_{mem}(L)$: "Given w; is $w \in L$?"
***membership problem** for L.* *w*

$P_{con}(\mathbb{D})$: "Given D_1, $D_2 \in \mathbb{D}$; is $L(D_1) \subseteq L(D_2)$?"
***containment problem** for $\mathbb{D}$.* *rep(D_1)#rep(D_2)*

$P_{ncon}(\mathbb{D})$: "Given D_1, $D_2 \in \mathbb{D}$; is $L(D_1) \not\subset L(D_2)$?"
***noncontainment problem** for $\mathbb{D}$.* *rep(D_1)#rep(D_2)*

$P_{eq}(\mathbb{D})$: "Given D_1, $D_2 \in \mathbb{D}$; is $L(D_1) = L(D_2)$?"
***equivalence problem** for $\mathbb{D}$.* *rep(D_1)#rep(D_2)*

$P_{neq}(\mathbb{D})$: "Given D_1, $D_2 \in \mathbb{D}$; is $L(D_1) \neq L(D_2)$?"
***inequivalence problem** for $\mathbb{D}$.* *rep(D_1)#rep(D_2)*

3.6 Applications to Lexical Analysis