Course 3

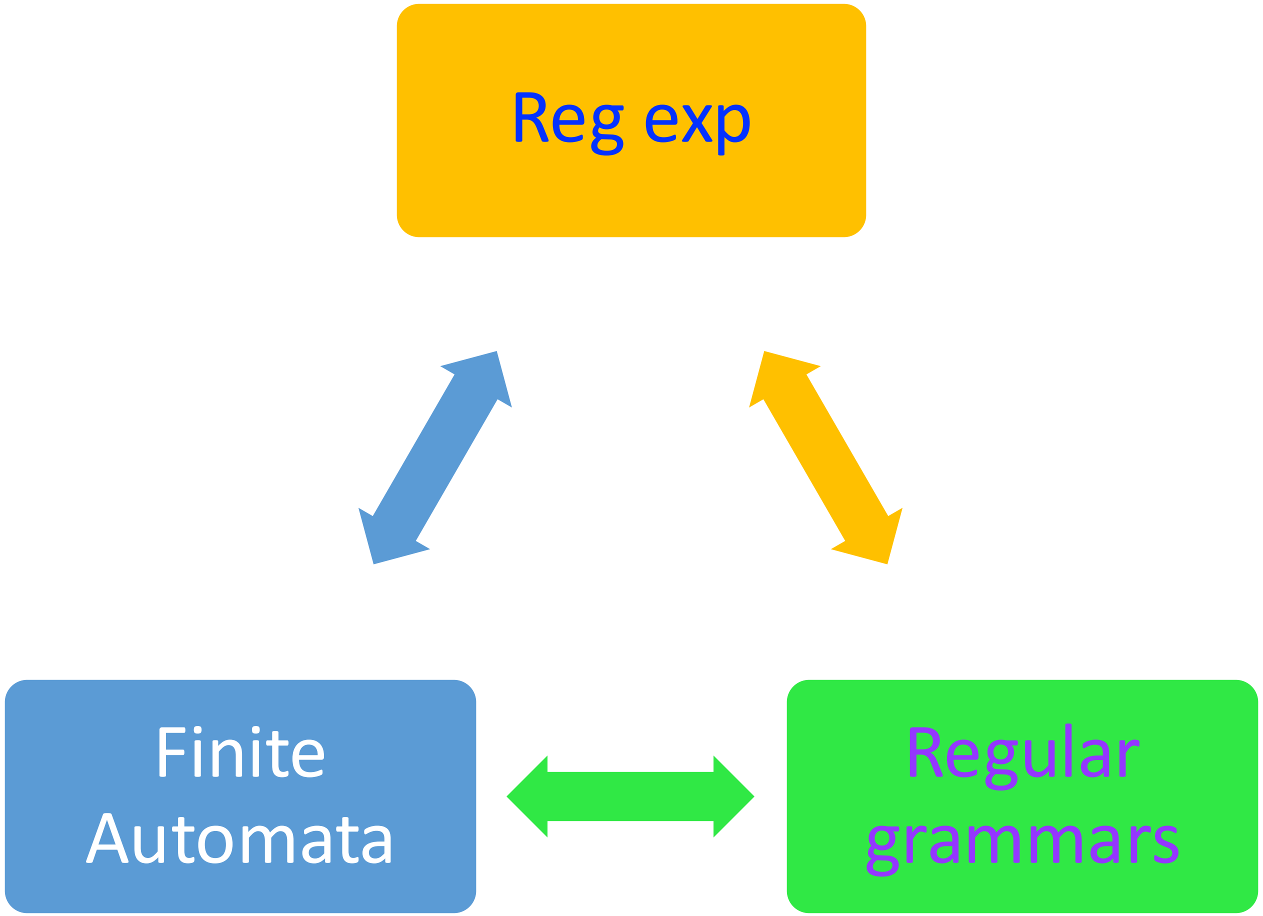
Formal Languages

- Basic notions -

Regular languages

Why?

1. Search engine – success of Google
2. Unix commands
3. Programming languages – new feature


Reg exp
Finite Automata
Regular grammars

Regular grammars

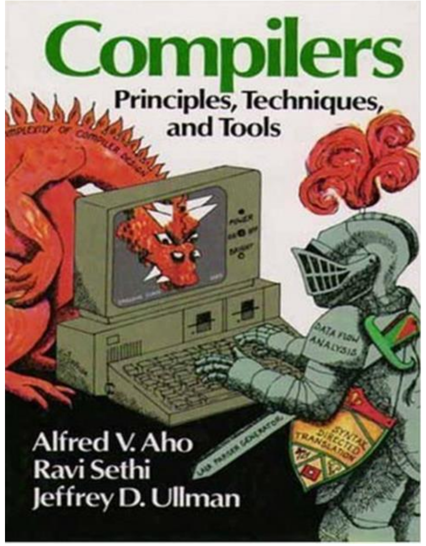
- G = (N, Σ, P, S) right linear grammar if

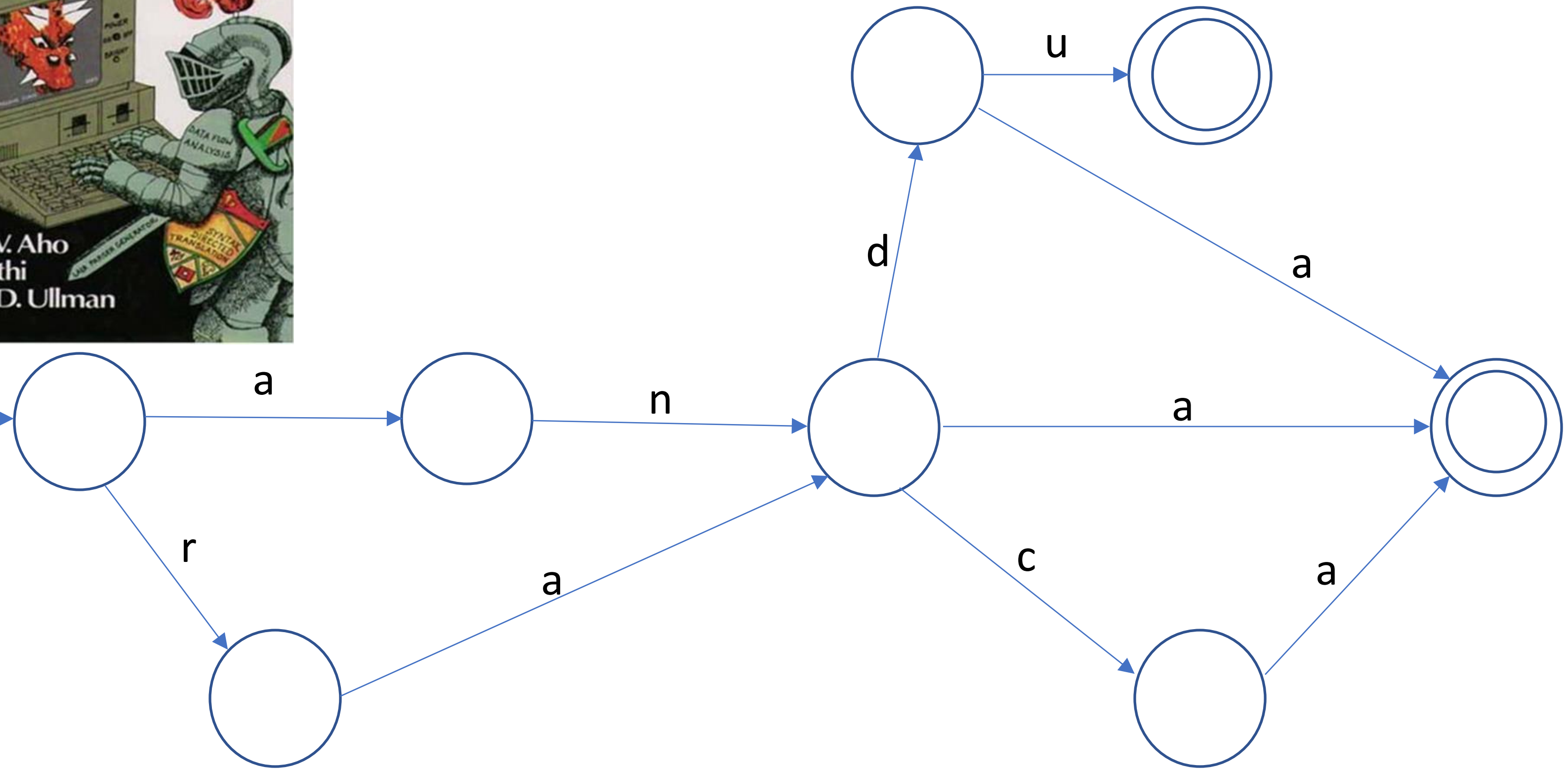
  $\forall p \in P$: $A \to aB$ or $A \to b$, where $A,B \in N$ and $a,b \in \Sigma$

- G = (N, Σ, P, S) regular grammar if
  - G is right linear grammar

  and
  - $A \to \varepsilon \notin P$, with the exception that $S \to \varepsilon \in P$, in which case S does not appear in the rhs (right hand side) of any other production

- $L(G) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow} w\}$ - right linear language

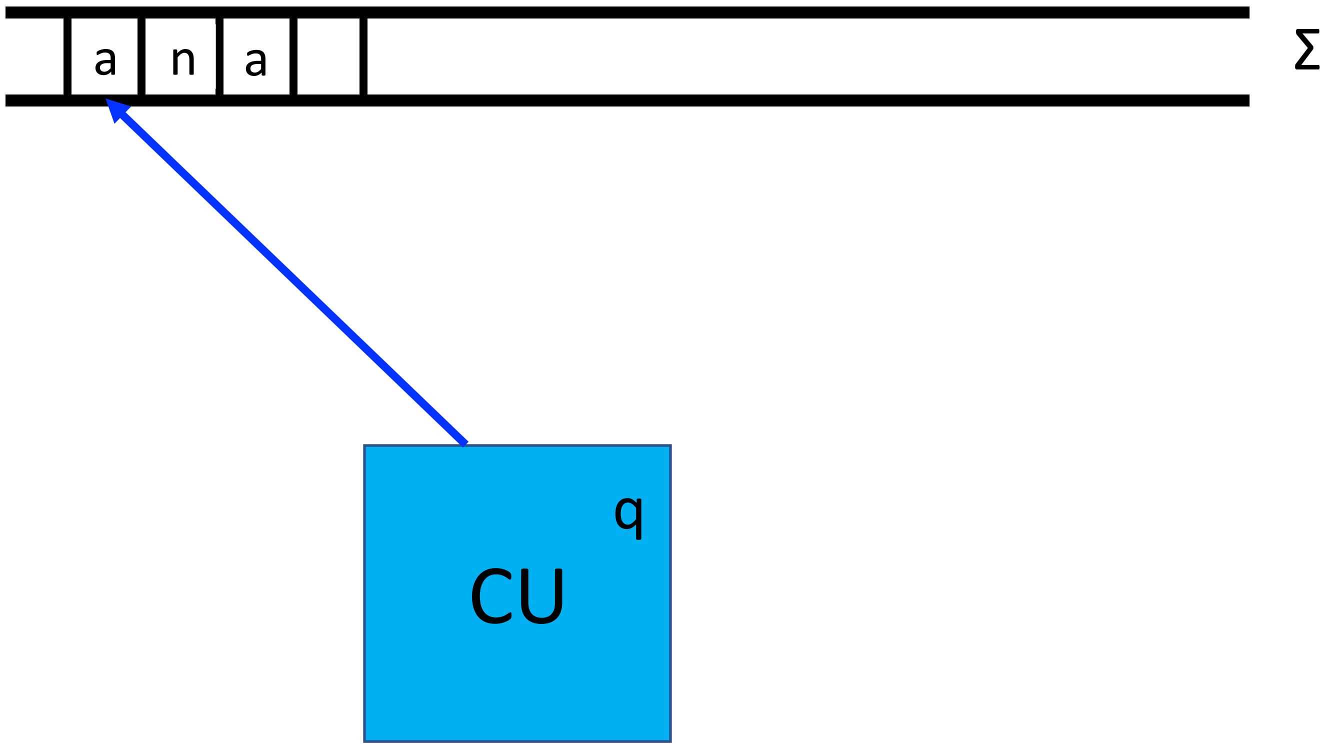




Problem: The door to the tower is closed by the Red Dragon, using a complicated machinery. Prince Charming has managed to steal the plans and is asking for your help. Can you help him determining all the person names that can unlock the door

Finite Automata

- Intuitive model



Definition: A ***finite automaton (FA)*** is a 5-tuple

$$M = (Q,\Sigma,\delta,q0,F)$$

where:

- Q - finite set of states ($|Q|<\infty$)
- Σ - finite alphabet ($|\Sigma|<\infty$)
- δ – transition function : $\delta:Q\times\Sigma\rightarrow P(Q)$
- q_0 – initial state $q_0 \in Q$
- $F\subseteq Q$ – set of final states

Remarks

1. $Q \cap \Sigma = \emptyset$
2. $\delta : Q \times \Sigma \rightarrow P(Q)$, $\varepsilon \in \Sigma^0$ - relation $\delta(q,\varepsilon)=p$ **NOT** allowed
3. If $|\delta(q,a)| \leq 1$ => deterministic finite automaton (DFA)
4. If $|\delta(q,a)| > 1$ (more than a state obtained as result) => nondeterministic finite automaton (NFA)

Property: For any NFA *M* there exists a DFA *M'* equivalent to *M*

Configuration $C=(q,x)$

where:

- q state
- x unread sequence from input: $x \in \Sigma^*$

Initial configuration : (q_0,w) , w - whole sequence

Final configuration: (q_f ,ε) , $q_f \in F$, ε –empty sequence

(corresponds to accept)

Relations between configurations

- $\vdash$ **move / transition** (simple, one step)
  $(q,ax) \vdash (p,x)$, $p \in \delta(q,a)$

- $\overset{k}{\vdash}$ **k move** = a sequence of k simple transitions) $C_0 \vdash C_1 \vdash \ldots \vdash C_k$
- $\overset{+}{\vdash}$ **+ move**
  $C \overset{+}{\vdash} C'$: $\exists$ $k>0$ such that $C \overset{k}{\vdash} C'$
- $\overset{*}{\vdash}$ *** move (star move)**
  $C \overset{*}{\vdash} C'$: $\exists$ $k \geq 0$ such that $C \overset{k}{\vdash} C'$

Definition : ***Language*** accepted by FA M = (Q,Σ,δ,q0,F) is:

$$L(M)=\{ w \in \Sigma^* \mid (q_0,w) \vdash^* (q_f ,\varepsilon) , q_f \in F \}$$

Remarks

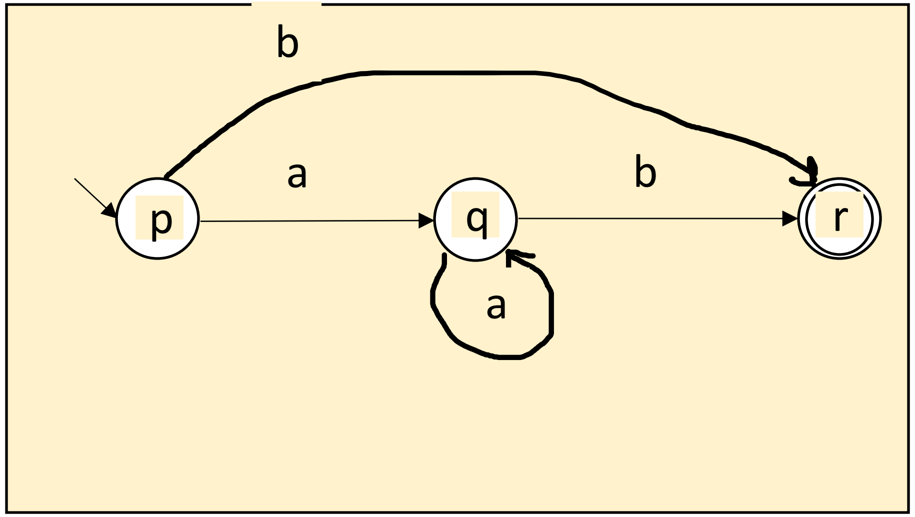
1. 2 finite automata M_1 and M_2 are equivalent if and only if they accept the same language

$$L(M_1)=L(M_2)$$

1. $\varepsilon \in L(M) \Leftrightarrow q_0 \in F$ (initial state is final state)

Representing FA

1. List of all elements
2. Table
3. Graphical representation



$M=(Q,\Sigma,\delta,p,F)$
$Q = \{p,q,r\}$
$\Sigma = \{a,b\}$
$\delta(p,a) = q$
$\delta(q,a)=q$
$\delta(q,b)=r$
$\delta(p,b)=r$
$F = \{r\}$

$M=(Q,\Sigma,\delta,p,F)$
$F = \{r\}$

	a	**b**
p	q	r
q	q	r
r	-	-

(p,aab)|-(q,ab)|-(q,b)|-(r,ε) => aab accepted
(p,aba)|-(q,ba)|-(r,a) => aba not accepted

Remember

- **Grammar**

$G=(N,\Sigma,P,S)$

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

- Finite automaton

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

$L(M)=\{ w \in \Sigma^* \mid (q_0,w) \vdash (q_f ,\varepsilon) , q_f \in F \}$

Theorem 1: For any regular grammar $G=(N, \boldsymbol{\Sigma}, P, S)$ there exists a FA $M=(Q, \boldsymbol{\Sigma}, \delta, q_0, F)$ such that $L(G) = L(M)$

Proof: construct M based on G

$Q = N \cup \{K\}, K \notin N$

$q_0 = S$

$F = \{K\} \cup \{S \mid \text{if } S \rightarrow \boldsymbol{\varepsilon} \in P\}$

δ: if $A \rightarrow aB \in P$ then $\delta(A,a) = B$

if $A \rightarrow a \in P$ then $\delta(A,a) = K$

Theorem 1: For any regular grammar $G=(N, \boldsymbol{\Sigma}, P, S)$ there exists a FA $M=(Q, \boldsymbol{\Sigma}, \delta, q_0, F)$ such that $L(G) = L(M)$

Proof: construct M based on G

$Q = N \cup \{K\}, K \notin N$

$q_0 = S$

$F = \{K\} \cup \{S \mid \text{if } S \to \boldsymbol{\varepsilon} \in P\}$

δ: if $A \to aB \in P$ then $\delta(A,a) = B$

if $A \to a \in P$ then $\delta(A,a) = K$

Prove that L(G) = L(M) ($w \in L(G) \Leftrightarrow w \in L(M)$):

$S \stackrel{*}{\Rightarrow} w \Leftrightarrow (S,w) \stackrel{*}{\vdash} (qf, \boldsymbol{\varepsilon})$

$w = \boldsymbol{\varepsilon}$: $S \stackrel{*}{\Rightarrow} \boldsymbol{\varepsilon} \Leftrightarrow (S, \boldsymbol{\varepsilon}) \stackrel{*}{\vdash} (S, \boldsymbol{\varepsilon})$ – true

$w = a_1 a_2 \ldots a_n$: $S \stackrel{*}{\Rightarrow} w \Leftrightarrow (S,w) \stackrel{*}{\vdash} (K, \boldsymbol{\varepsilon})$

$S \Rightarrow a_1A_1 \Rightarrow a_1a_2A_2 \Rightarrow \ldots \Rightarrow a_1a_2\ldots a_{n-1}A_{n-1} \Rightarrow a_1a_2\ldots a_{n-1}a_n$

$S \Rightarrow a_1A_1$ exists if $S \to a_1A_1$ and then $\delta(S,a_1)=A_1$

$A_1 \to a_2A_2 : \delta(A_1,a_2)=A_2 \ldots$

$A_{n-1} \to a_n : \delta(A_{n-1},a_n)=K$

$(S,a_1a_2\ldots a_n) \vdash (A_1,a_2\ldots a_n) \vdash (A_2,a_3\ldots a_n) \vdash \ldots \vdash (A_{n-1},a_n) \vdash (K, \boldsymbol{\varepsilon})$, $K \in F$

Theorem 2: For any FA $M=(Q, \mathbf{\Sigma}, \delta, q_0, F)$ there exists a right linear grammar $G=(N, \mathbf{\Sigma}, P, S)$ such that $L(G) = L(M)$

Proof: construct G based on M

$N = Q$

$S = q_0$

P: if $\delta(q,a) = p$ then $q \rightarrow ap \in P$

if $p \in F$ then $q \rightarrow a \in P$

if $q_0 \in F$ then $S \rightarrow \boldsymbol{\varepsilon}$

Theorem 2: For any FA $M=(Q, \boldsymbol{\Sigma}, \delta, q_0,F)$ there exists a right linear grammar $G=(N, \boldsymbol{\Sigma}, P, S)$ such that $L(G) = L(M)$

Proof: construct G based on M

$N = Q$

$S = q_0$

P: if $\delta(q,a) = p$ then $q \rightarrow ap \in P$

if $p \in F$ then $q \rightarrow a \in P$

if $q_0 \in F$ then $S \rightarrow \boldsymbol{\varepsilon}$

Prove that L(M) = L(G) *($w \in L(M) \Leftrightarrow w \in L(G)$):*

P(i): $q \overset{i+1}{\Rightarrow} x \Leftrightarrow (q,x) \overset{i}{\vdash} (q_f , \boldsymbol{\varepsilon})$, $q_f \in F$ -prove by induction

Apply P : $q_0 \overset{i+1}{\Rightarrow} w \Leftrightarrow (q_0,w) \overset{i}{\vdash} (q_f , \boldsymbol{\varepsilon})$, $q_f \in F$

If i=0: $q \Rightarrow x \Leftrightarrow (q,x) \overset{0}{\vdash} (q_f , \boldsymbol{\varepsilon})$ $(x= \boldsymbol{\varepsilon},q=q_f)$ $q \Rightarrow \boldsymbol{\varepsilon} \Leftrightarrow q_0 \rightarrow \boldsymbol{\varepsilon}$, $q_0 \in F$

Assume $\forall$ $k \leq i$ P is true

$q \overset{i+1}{\Rightarrow} x \Leftrightarrow (q,x) \overset{i}{\vdash} (q_f , \boldsymbol{\varepsilon})$

For $q \in N$ apply "$\Rightarrow$" : $q \Rightarrow ap \overset{i}{\Rightarrow} ax$

If $q \Rightarrow ap$ then $\delta(q,a)= p$; if $p \overset{i}{\Rightarrow} ax$ then $(p,x) \overset{i-1}{\vdash} (q_f , \boldsymbol{\varepsilon})$, $qf \in F$

THEN $(q,ax) \overset{i}{\vdash} (q_f , \boldsymbol{\varepsilon})$, $qf \in F$

Regular sets

Definition: Let Σ be a finite alphabet. We define regular sets over Σ recursively in the following way:

1. $\boldsymbol{\Phi}$ is a regular set over Σ (empty set)
2. $\{\boldsymbol{\varepsilon}\}$ is a regular set over Σ
3. $\{a\}$ is a regular set over Σ, $\forall\ a \in \Sigma$
4. If P, Q are regular sets over Σ, then $P \cup Q$, PQ, P^* are regular sets over Σ
5. Nothing else is a regular set over Σ

Regular expressions

Definition: Let $\boldsymbol{\Sigma}$ be a finite alphabet. We define regular expressions over $\boldsymbol{\Sigma}$ recursively in the following way:

1. $\boldsymbol{\Phi}$ is a regular expression denoting the regular set $\boldsymbol{\Phi}$ (empty set)
2. $\boldsymbol{\varepsilon}$ is a regular expression denoting the regular set $\{\boldsymbol{\varepsilon}\}$
3. **a** is a regular expression denoting the regular set {a}, $\forall\ a \in \boldsymbol{\Sigma}$
4. If **p,q** are regular expression denoting the regular sets P, Q then:
   - **p+q** is a regular expression denoting the regular set $P \cup Q$,
   - **pq** is a regular expression denoting the regular set PQ,
   - **p*** is a regular expression denoting the regular set P^*
5. Nothing else is a regular expression

Remarks:

Examples

1. $p^+ = pp^*$
2. Use paranthesis to avoid ambiguity
3. Priority of operations: *, concat, + (from high to low)
4. For each regular set we can find at least one regular exp to denote it (there is an infinity of reg exp denoting them)
5. For each regular exp, we can construct the corresponding regular set
6. 2 regular expressions are **equivalent** **iff** they denote the same regular set

Algebraic properties of regular exp

Let $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$ be regular expressions.

1. $\alpha + \beta = \beta + \alpha$
2. $\Phi^* = \varepsilon$
3. $\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma$
4. $\alpha(\beta\gamma) = (\alpha\beta)\gamma$
5. $\alpha (\beta + \gamma) = \alpha\beta + \alpha\gamma$
6. $(\alpha + \beta)\gamma = \alpha\gamma + \beta\gamma$
7. $\alpha \varepsilon = \varepsilon \alpha = \alpha$
8. $\Phi\alpha = \alpha\Phi = \Phi$
9. $\alpha^* = \alpha + \alpha^*$
10. $(\alpha^*)^* = \alpha^*$
11. $\alpha + \alpha = \alpha$
12. $\alpha + \Phi = \alpha$

Reg exp equations

- Normal form: **X = aX + b**

  where a,b – reg exp

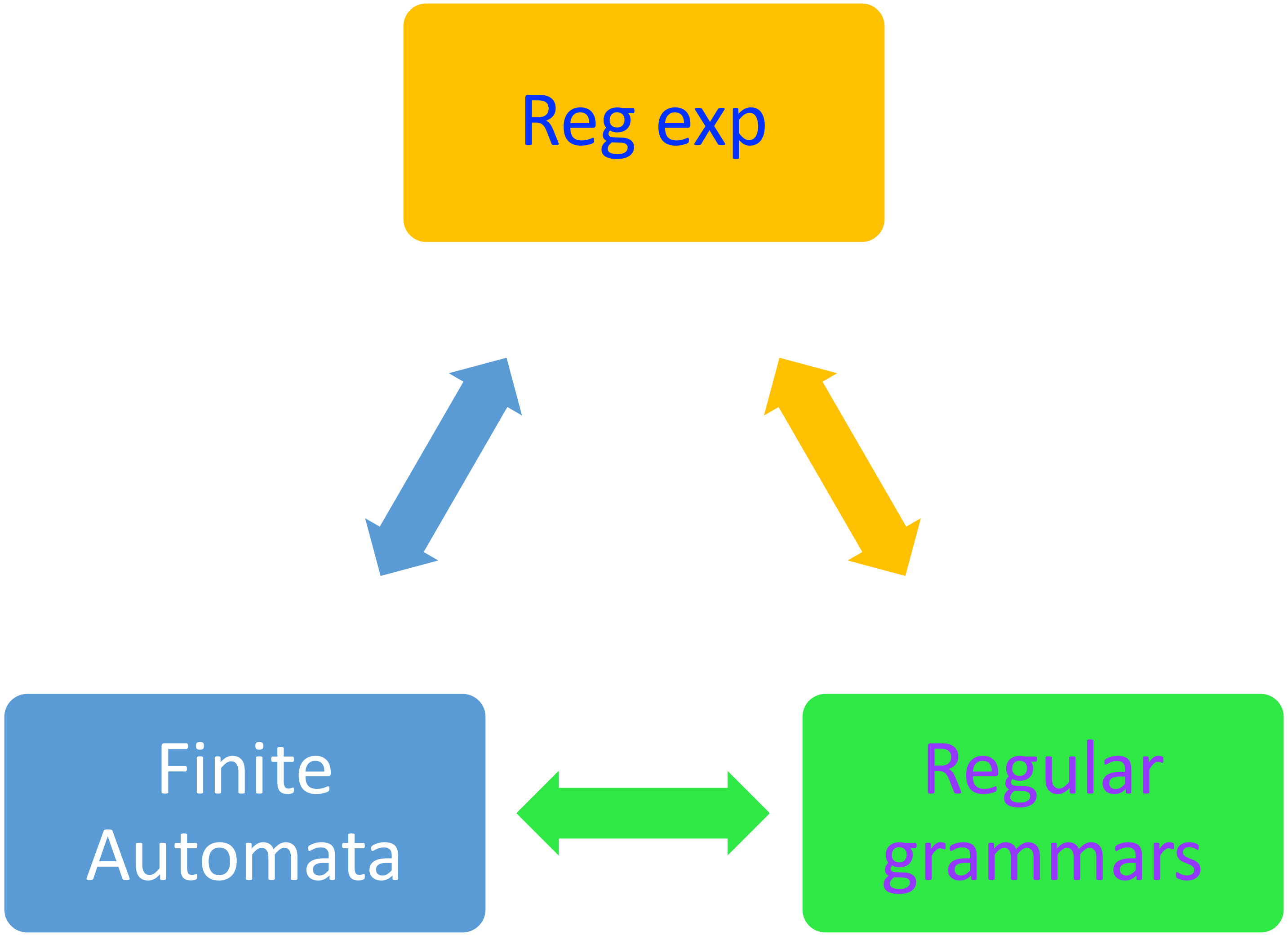
- Solution: **X = a*b**

a a*b + b = (aa* +**ε**)b = a*b

- System of reg exp equations:

$$\begin{cases} X = a_1 X + a_2 Y + a_3 \\ Y = b_1 X + b_2 Y + b_3 \end{cases}$$

- Solution: Gauss method (replace X_i and solve X_n)


Reg exp
Finite Automata
Regular grammars

Prop: *Regular sets are right linear languages*

Lemma 1: $\boldsymbol{\Phi}$,{$\boldsymbol{\varepsilon}$}, {a},∀a∈$\boldsymbol{\Sigma}$ are right linear languages

Proof: constructive

i. G = ({S}, $\boldsymbol{\Sigma}$, $\boldsymbol{\Phi}$, S) – regular grammar such that L(G) = $\boldsymbol{\Phi}$

ii. G = ({S}, $\boldsymbol{\Sigma}$,{S→$\boldsymbol{\varepsilon}$}, S) – regular grammar such that L(G) ={$\boldsymbol{\varepsilon}$}

iii. G = ({S}, $\boldsymbol{\Sigma}$,{S→a}, S) – regular grammar such that L(G) ={a}

Lemma 2: If L_1 and L_2 are right linear languages then:

$L_1 \cup L_2$, L_1L_2 and L_1^* are right linear languages.

Proof: constructive

L_1, L_2 right linear languages => $\exists G_1$, G_2 such that

$G_1 = (N_1, \Sigma_1, P_1, S_1)$ and $L_1 = L(G_1)$

$G_2 = (N_2, \Sigma_2, P_2, S_2)$ and $L_2 = L(G_2)$ assume $N_1 \cap N_2 = \emptyset$

i. $G_3 = (N_3, \boldsymbol{\Sigma}, P_3, S_3)$

$N_3 = N_1 \cup N_2 \cup \{S_3\}$; $\Sigma_3 = \Sigma_1 \cup \Sigma_2$

$P_3 = P_1 \cup P_2 \cup \cancel{\{S_3 \rightarrow S_1 \mid S_2\}}$

$\{S_3 \rightarrow \boldsymbol{\alpha}_1 \mid S_1 \rightarrow \boldsymbol{\alpha}_1 \in P_1\} \cup \{S_3 \rightarrow \boldsymbol{\alpha}_2 \mid S_2 \rightarrow \boldsymbol{\alpha}_2 \in P_2\}$

G_3 – right linear language

and

$L(G_3) = L(G_1) \cup L(G_2)$

PROOF!!! Homework

ii. $G_4 = (N_4, \boldsymbol{\Sigma}, P_4, S_4)$

$N_4 = N_1 \cup N_2$; $S_4 = S_1$; $\Sigma_4 = \Sigma_1 \cup \Sigma_2$

$$
\begin{aligned}
P_4 = \quad & \{A \to aB \mid \text{if } A \to aB \in P_1\} \cup \\
& \{A \to aS_2 \mid \text{if } A \to a \in P_1\} \cup \\
& P_2 \cup \\
& \{S_1 \to \boldsymbol{\alpha}_2 \mid \text{if } S_1 \to \boldsymbol{\varepsilon} \in P_1 \text{ and } S_2 \to \boldsymbol{\alpha}_2 \in P_2\}
\end{aligned}
$$

G_4 – right linear language

and

$\mathbf{L(G_4) = L(G_1)\, L(G_2)}$

PROOF!!! Homework

iii. $G_5 = (N_5, \boldsymbol{\Sigma}_1, P_5, S_5)$

//IDEA: concatenate L_1 with itself

$N_4 = N_1 U \{S_5\}$;

$P_5 = \quad P_1 \text{ U } \{S_5 \rightarrow \boldsymbol{\varepsilon}\} \text{ U}$

$\{S_5 \rightarrow \boldsymbol{\alpha}_1 \mid S_1 \rightarrow \boldsymbol{\alpha}\,_1 \in P_1\} \text{ U}$

$\{A \rightarrow aS_1 \mid \text{if } A \rightarrow a \in P_1\}$

G_5 – right linear language

and

$L(G_5) = L(G_1)^*$

PROOF!!! Homework