WEEK-5 : Lecture Notes

Theorem (DFA to regular expression)

If L is accepted by a DFA, then L is denoted by a regular expression

Proof:

Let, L : set accepted by the DFA

$M: \{\{q_1, \dots q_n\}, \Sigma, \delta, q_1, F)$.

- R_{ij}^{k} - set of all strings x s.t.

$$\delta(q_i, x) = q_j$$

and if $\delta(q_i, y) = q_l$ and y is a prefix of x other than x or ε then $l \le k$

where is an intermediate node on the path from q_i to q_j when the path has label x.

- for $R_{i,j}^{k}$, i,j may be $> k$

- $R_{i,j}^{n}$ - all strings that take q_i to q_j and no states has number greater than $\underline{n}$

  - no restriction at all on the paths represented as no states with index greater than $\underline{n}$.

- We can define $R_{i,j}^{k}$ recursively

$$R_{i,j}^{k} = R_{i,j}^{k-1} \cup R_{i,k}^{k-1} \left(R_{kk}^{k-1}\right)^{*} R_{k,j}^{k-1}$$

$$R_{i,j}^{0} = \begin{cases} \{a \mid \delta(q_i, a) = q_j\} & \text{if } i \neq j \\ \{a \mid \delta(q_i, a) = q_j\} \cup \{\varepsilon\} & \text{if } i = j \end{cases}$$

Informally $R_{i,j}^{k}$ means

$$q_i \xrightarrow{\text{no state higher than } q_k} q_j$$

i.e. if q_l is an intermediate node on this path $l \leq k$

Two possibilities:

- $R_{i,j}^{k} = R_{i,j}^{k-1}$ The path doesnot go through q_k at all, i.e. label of the path is in $R_{i,j}^{k-1}$

- $R_{i,j}^{k} = R_{i,k}^{k-1} \cdots$ The path goes through q_k at least once.

$$q_i \to q_k \to q_k \to q_k \to q_k \to q_j$$

no state higher than q_k i.e. label of this part is in R_{ik}^{k-1}

without passing through q_k or a higher-numbered state

zero or more number of states / parts, i.e. label in $\left(R_{kk}^{k-1}\right)^{*}$

no state higher than q_k i.e. label ~~in~~ of this part is in R_{kj}^{k}

Claim: For each i, j, k, there exists a regular expression r_{ij}^{k} denoting the language R_{ij}^{k}

Proof: (by induction on k)

Base $(k = 0)$

- $R_{i,j}^{0}$ = a finite set of strings each of which is either ε or a single symbol.

$$\therefore\ r_{i,j}^{0} = \begin{cases} a_1 + a_2 + \dots + a_p \ (\text{or } a_1 + a_2 + \dots + a_p + \varepsilon \text{ if } i = j) \\ \quad \text{when } \delta(q_i, a) = q_j \\ \qquad \text{for } a \in \{a_1, a_2 \dots, a_p\} \\ \phi \ (\text{or } \varepsilon \text{ if } i = j) \text{ otherwise.} \end{cases}$$

- $L(r_{i,j}^{0}) = R_{i,j}^{0}$

Induction:

By induction hypothesis, for each l, m, $\exists$ a regular expression r_{lm}^{k-1} s.t. $L(r_{l,m}^{k-1}) = R_{l,m}^{k-1}$

Now for $r_{i,j}^{k}$, we may select the regular expression

$$r_{i,j}^{k-1} + r_{i,k}^{k-1} \left(r_{kk}^{k-1} \right)^{*} r_{kj}^{k-1}$$

which completes the induction:

$$L(r_{i,j}^{k}) = L(r_{i,j}^{k-1}) \cup L(r_{ik}^{k-1})\, L(r_{kk}^{k-1})^{*}$$

$$R_{ij}^{k} = R_{ij}^{k-1} \cup R_{ik}^{k-1} (R_{kk}^{k-1})^{*} R_{kj}^{k-1}$$

Now observe that

$$L = L(M) = \bigcup_{q_j \in F} R_{1j}^{n}$$

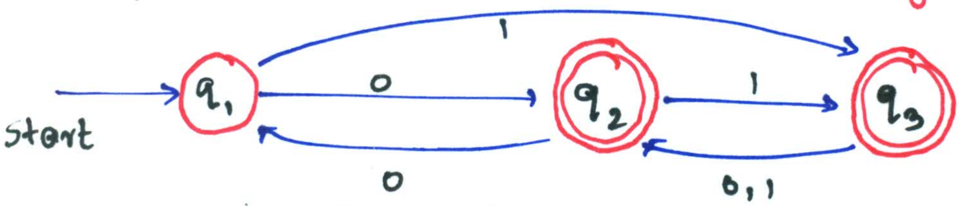
as R_{1j}^{n} denotes the labels of all paths from q_1 to an accepting state q_j of F

Then $L = L(M)$ is denoted by the regular expression

$$r_{1j_1}^{n} + r_{1j_2}^{n} + \cdots + r_{1j_m}^{n}$$

when $F = \{q_{j_1}, q_{j_2}, \ldots, q_{j_m}\}$

Example (Conversion of DFA to regular expression)



$n = 3$

$F = \{q_2, q_3\}$

To find $r_{12}^{3} + r_{13}^{3}$

	$K=0$	$K=1$	$K=2$
r_{11}^{K}	ε	ε	$(00)^*$
r_{12}^{K}	0	0	$0(00)^*$
r_{13}^{K}	1	1	0^*1
r_{21}^{K}	0	0	$0(00)^*$
r_{22}^{K}	ε	$\varepsilon+00$	$(00)^*$
r_{23}^{K}	1	$1+01$	0^*1
r_{31}^{K}	ϕ	ϕ	$(0+1)(00)^*0$
r_{32}^{K}	$0+1$	$0+1$	$(0+1)(00)^*$
r_{33}^{K}	ε	ε	$\varepsilon+(0+1)0^*1$

Closure properties of Regular languages / sets

Regular sets are closed under:

- union
- concatenation
- Klune closure

(the three above:) Immediate from the definition of regular expression

- complementation
- intersection ← $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$
- Reversal
- substitution
- homomorphism
- inverse homomorphism

Direct construction of a DFA for intersection of two regular sets.

Consider the DFAs

$M_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1)$ $\quad M_2 = (Q_2, \Sigma_2, \delta_2, q_2, F_2)$

$$M = (Q_1 \times Q_2, \Sigma = \Sigma_1 \cup \Sigma_2, \delta, (q_1, q_2), F_1 \times F_2)$$

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

$\forall\, p_1 \in Q_1, p_2 \in Q_2, a \in \Sigma$

It is easy to show that

$$L(M) = L(M_1) \cap L(M_2).$$

Proof:

By induction of $|w|$, one can show that

$$\hat{\delta}((q_1, q_2), w) = (\hat{\delta}_1(q_1, w), \hat{\delta}_2(q_2, w))$$

where $w \in \Sigma^*$

Now, M accepts w iff $\hat{\delta}((q_1, q_2), w) \in F_1 \times F_2$

i.e. iff $\hat{\delta}_1(q_1, w) \in F_1$ $\hat{\delta}_2(q_2, w) \in F_2$

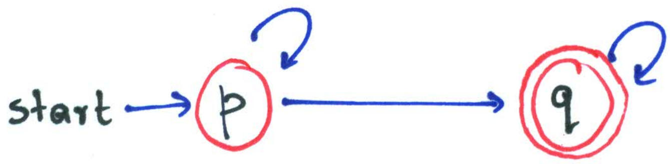
$\Rightarrow$ $w \in L(M)$ iff $w \in L(M_1)$ and $w \in L(M_2)$

i.e. iff $w \in L(M_1) \cap L(M_2)$

$\Rightarrow$ $L(M) = L(M_1) \cap L(M_2)$

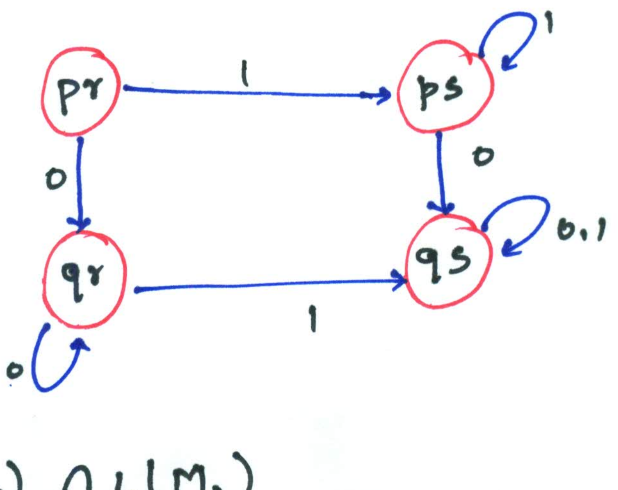
Example:

M_1 >



M_2 >

M >

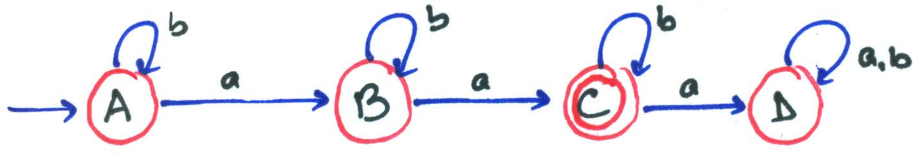


$L(M) = L(M_1) \cap L(M_2)$

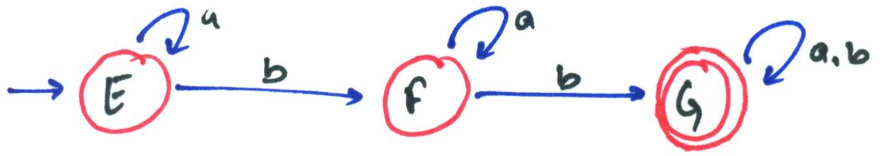
Example: DFA for {w | w has exactly two a's and atleast two b's}

Solution:

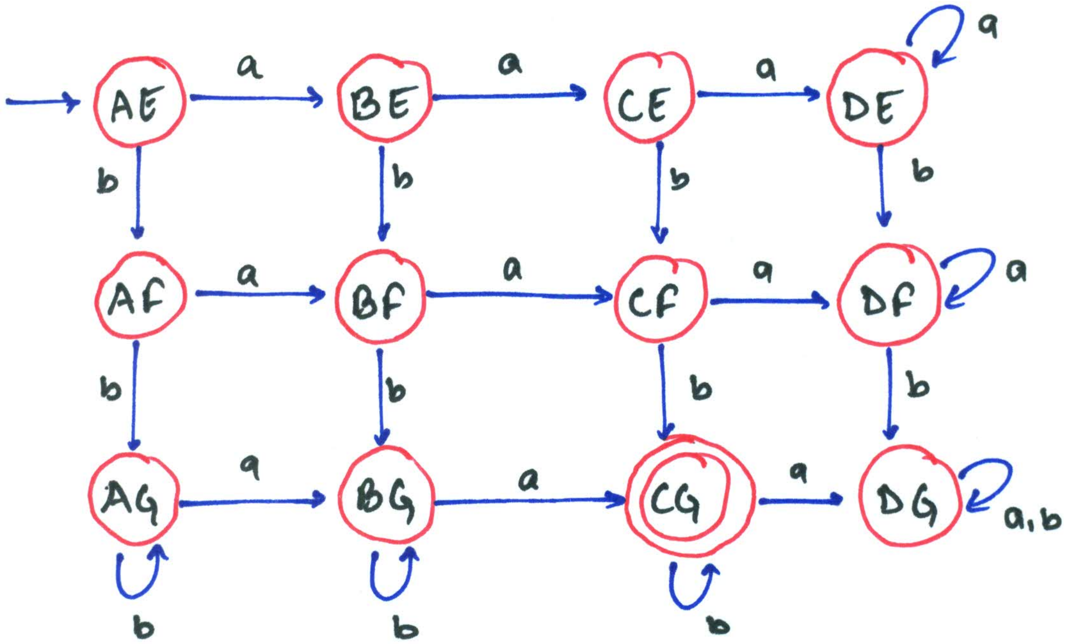
DFA for {w | w has exactly two a's}



DFA for {w | w has atleast two b's



Intersection

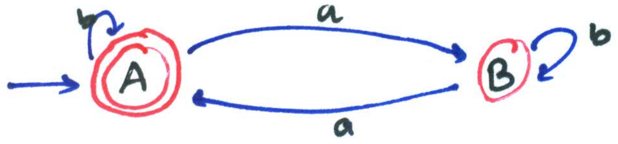


Example:

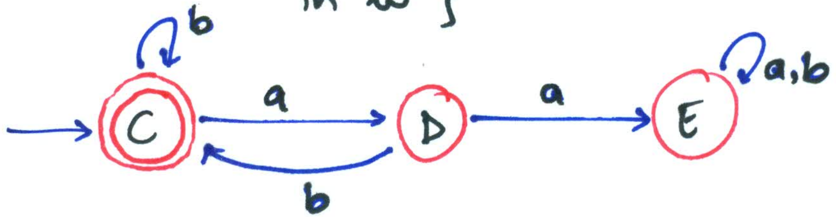
DFA for {w | w has an even number of a's and each a is followed by atleast one b}

Soln:

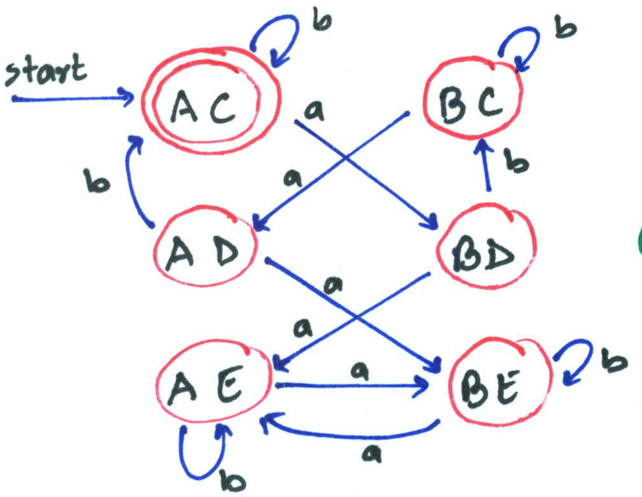
DFA for {w | w has an even no. of a's}



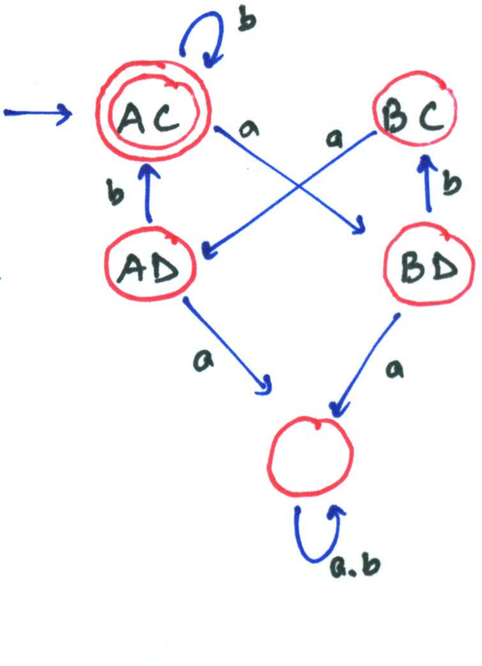
DFA for { w | each a is followed by atleast one b in w }



Combining them using the intersection construction given the DFA



OR



Theorem:

If L is a regular language over Σ, then $\bar{L} = \Sigma^* - L$ is also a regular language

Given a regular expression that recognizes L over Σ find another expression that defines the complement language

$$\begin{array}{ccccc} \text{regular expression} & \longrightarrow & \varepsilon\text{-NFA} & \xrightarrow{\text{subset construction}} & \text{DFA} \\ & & & & \downarrow \\ \text{regular expression} & \longleftarrow & \text{DFA} & \xleftarrow[\text{complement the accepting states.}]{} & \end{array}$$

Proof:

- Let M be a DFA such that $L(M) = L$

  $M = (Q, \Sigma_1, \delta, q_0, F)$

- $L \subseteq \Sigma^*$
- We may assume $\Sigma_1 = \Sigma$ as
  - if there are symbols in Σ_1 not in Σ then we may delete all transitions of M on symbols not in Σ
  - It will not change $L(M)$ as $L \subseteq \Sigma^*$
  - if there are symbols in Σ not in Σ_1, then none of these symbols appear in the set $L(M)$ as M is over the alphabet Σ_1

We introduce a dead state d into M with

$$\delta(d, a) = d \quad \forall\, a \in \Sigma$$

$$\delta(q, a) = d \quad \forall\, a \in \Sigma - \Sigma_1$$

- DFA $M = (Q, \Sigma, \delta, q_0, F)$ such that $L = L(M)$

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

Then $L(M') = \overline{L}$ as M' accepts a string $w \in \Sigma^*$

iff $\hat{\delta}(q_0, w) \in Q - F$

i.e. iff $\hat{\delta}(q_0, w) \notin F$

i.e. iff $w \notin L = L(M)$

i.e. iff $w \in \Sigma^* - L = \overline{L}$

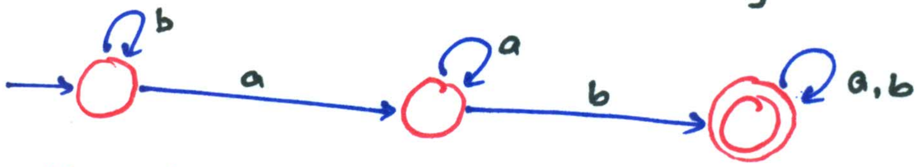
Note:

- M should be deterministic and without ε-moves
- $\{0,1\}^*$ regular set and ϕ is regular set over the alphabet $\Sigma = \{0,1\}$

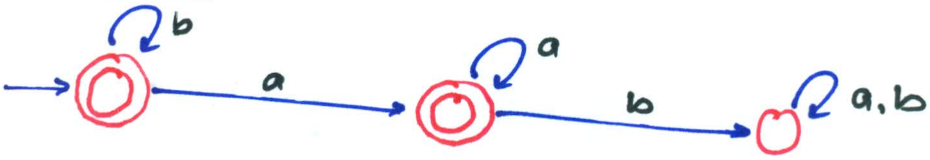
Example: DFA for {w| w doesnot contain the substring ab}

Solution:

DFA for {w| w contains ab}



- compliment

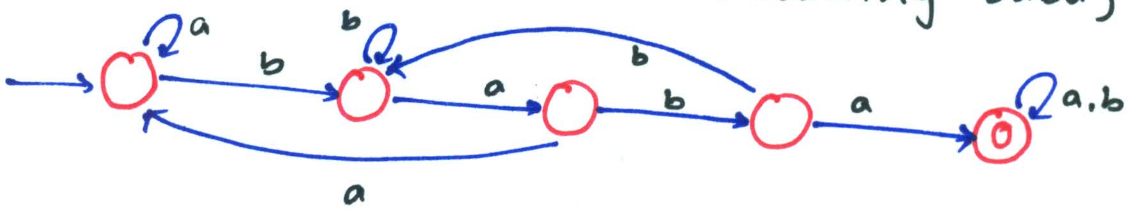


Example:

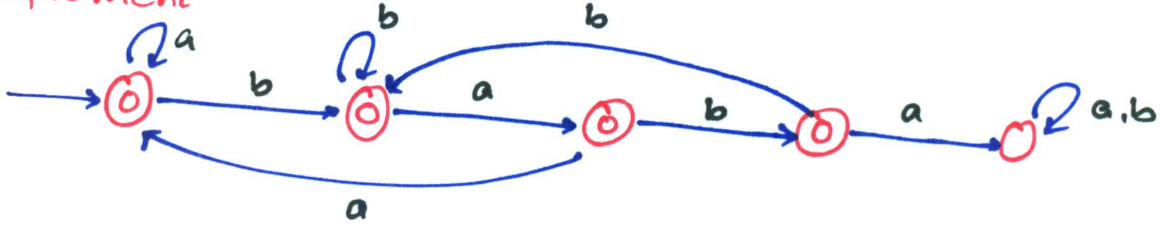
DFA for L = {w| w doesnot contain the substring baba}

Solution:

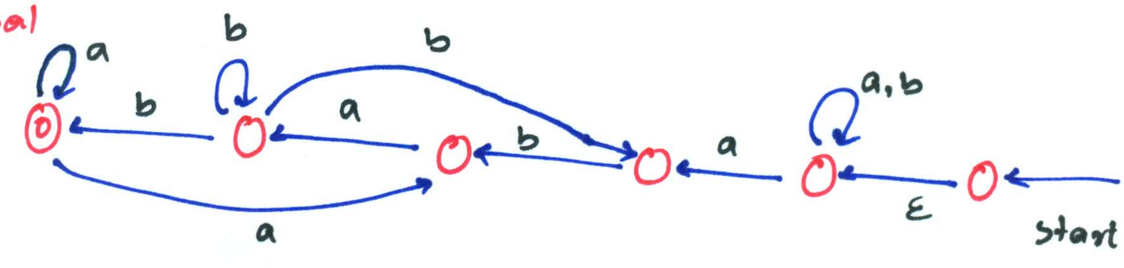
DFA for {w| w contains the substring baba}



- complement



reversal



Reversal

reversal of a string $a_1 a_2 \cdots a_n \rightarrow a_n a_{n-1} \cdots a_1$

w^R reversal of string w, e.g. $0010^R = 0100$

$\varepsilon^R = \varepsilon$

reversal of language $L \rightarrow L^R$

$L = \{001, 10, 111\} \rightarrow L^R = \{100, 01, 111\}$

Theorem:

If L is a regular language, so is L^R

Proof:

Let M be a ε-NFA that recognizes L

i.e. $L = L(M)$

Construct an ε-NFA, M' from M by:

1. Reverse all the arcs in the transition diagram of M
2. Make the start state of M to be the only accepting state for M'
3. Create a new start state p_0 with transition on ε to all the accepting states of M

M' simulates M "in reverse" and accepts w iff M accepts w^R

Formal proof of the reversal theorem using regular expressions:

Theorem: If L is a regular language, so is L^R

Proof:

Let L be defined by regular expressions, i.e. $L = L(s)$

Claim: $\exists$ another regular expression s^R s.t. $L(s^R) = (L(s))^R$

i.e. language of the regular expression s^R is the reversal of language of s.

We prove it by structural induction on size of regular expression s

Basis:

if $s = \varepsilon$, ϕ or a for some symbol $a \in \Sigma$

then $s^R = \varepsilon$, ϕ or a respectively

i.e. $\{\varepsilon\}^R = \{\varepsilon\}$, $\phi^R = \phi$, $\{a\}^R = \{a\}$

i.e. $(L(s))^R = L(s^R)$

Induction:

Case I:

$$S = S_1 + S_2 \Rightarrow S^R = S_1^R + S_2^R$$

e.g. $L(S_1) = \{01, 111\}$, $(L(S_1))^R = \{10, 111\}$

$L(S_2) = \{00, 10\}$, $(L(S_2))^R = \{00, 01\}$

$L(S) = L(S_1) \cup L(S_2) = \{01, 111, 00, 10\}$

$(L(S))^R = \{10, 111, 00, 01\}$

$(L(S_1))^R \cup (L(S_2))^R = \{10, 111, 00, 10\}$

$= L(S_1) \cup L(S_2)$

$= (L(S))^R$

$L(S_1^R) \cup L(S_2^R) = L(S_1^R + S_2^R) = L(S^R)$.

Case II:

$$S = S_1 S_2 \Rightarrow S^R = S_2^R S_1^R$$

$L(S_1 S_2) = \{0100, 0110, 11100, 11110\}$

$\Rightarrow (L(S_1 S_2))^R = \{0010, 0110, 00111, 01111\}$

$(L(S_2))^R = \{00, 01\}$, $(L(S_1))^R = \{10, 111\}$

$\Rightarrow (L(S_2))^R (L(S_1))^R = \{0010, 00111, 0110, 01111\}$

$= (L(S_1 S_2))^R$

In general

if $w \in L(S)$ where $w = w_1 w_2$, $w_1 \in L(S_1)$, $w_2 \in L(S_2)$

the $w^R = w_2^R w_1^R$

Case III

$s = s_1^* \Rightarrow s^R = (s_1^R)^*$

Let $w \in L(s)$ has the form $w = w_1 w_2 \dots w_n$ where each $w_i \in L(s_1)$

Then, $w^R = w_n^R w_{n-1}^R \dots w_1^R$, where each $w_i^R \in L(s_1^R)$

$\Rightarrow w^R \in L((s_1^R)^*)$

Conversely any string w in $L((s_1^R)^*)$ is of the form $w_1 w_2 \dots w_n$, when each w_i is the reversal of a string in $L(s_1)$

$\Rightarrow w^R = w_n^R w_{n-1}^R \dots w_1^R \in L(s_1^*) = L(s)$

Thus we have shown that $w \in L(s)$ iff it's reversal $w^R \in L((s_1^R)^*)$

This establishes thee fact that

$s^R = (s_1^R)^*$ if $s = s_1^*$

Example:

Let L be defined by a regular expression $(0+1)0^*$

Then L^R is the language of

$(0^*)^R (0+1)^R$

i.e. $(0^R)^* (0^R + 1^R)$

i.e. $0^* (0+1)$