Closure properties (cont.)

Pumping Lemma

Ex:

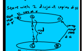
Def: Given $L \subseteq \Sigma^*$ and $w \in \Sigma^*$, we define

$\mathrm{Insert}_w(L) := \{xwy : xy \in L\}$

Prop: If L is regular, then $\mathrm{Insert}_w(L)$ is regular (for fixed w)

Proof 1: Let N be a clean NFA recog. L. We construct NFA N' recog. $\mathrm{Insert}_w(L)$ as follows.

Start with 2 disjoint copies of N



Add w-path from state of N_1 to corresp. state of N_2, for every state of N_1. //

Proof 2: Given a regex r for L, we make a regex r' for $\mathrm{Insert}_w(L)$ according to these rules:

	r	r'
	$\emptyset$	$\emptyset$
$(a \in \Sigma)$	a	$aw \cup wa$
s,t regexes	$s \cup t$	$s' \cup t'$
	st	$s't \cup st'$
	s^*	$\underline{s^*s's^*} \cup w$

Def: Given $L \subseteq \Sigma^*$, define

$L/a := \{w \in \Sigma^* : wa \in L\}$

$(a \in \Sigma$ fixed$)$

Prop: If L is reg then L/a is reg.

Proof: Given a DFA A for L we construct a DFA A' for L/a:

Let $A = \langle Q, \Sigma, \delta, q_0, F \rangle$

$A' := \langle Q, \Sigma, \delta, q_0, F' \rangle$

where

$F' := \{q \in Q : \delta(q, a) \in F\}$

(Omitting proof of correctness — just string & bisimulation) //

Proof alt: Given regex r for L, construct r' for L/a as follows:

	r	r'
literal	a	$\varepsilon \; (= \emptyset^*)$
	$\emptyset$	$\emptyset$
$(b \in \Sigma, b \neq a)$	b	$\emptyset$
	$s \cup t$	$s' \cup t'$
	st	$\begin{cases} st' \cup s' & \text{if } \varepsilon \in t \\ st' & \text{otherwise} \end{cases}$
	s^*	s^*s'

//

Let $w, x \in \Sigma^*$

A merge of w & x is a string obtained by merging symbols from w & x in some arbitrary way:

$w_1 x_1 w_2 x_2 \cdots w_k x_k$

where $w_1, \ldots, w_k, x_1, \ldots, x_k \in \Sigma^*$ and

$w = w_1 \cdots w_k$

$x = x_1 \cdots x_k$

$\mathrm{Merge}(x, y) := \{z : z \text{ is a merge of } x \text{ and } y\}$

$\Sigma = \{a, b, c\}$

$\mathrm{merge}(ab, ac) = \{abac, aabc, acab, aacb, \ldots\}$

Given Language L_1, L_2, define

$\mathrm{Merge}(L_1, L_2) := \bigcup_{x \in L_1, y \in L_2} \mathrm{Merge}(x, y)$

Prop: If L_1 & L_2 are regular, then $\mathrm{Merge}(L_1, L_2)$ is regular.

Proof: Let $A_1 = \langle Q_1, \Sigma, \delta_1, q_1, F_1 \rangle$

$A_2 = \langle Q_2, \Sigma, \delta_2, q_2, F_2 \rangle$

be DFAs recog. L_1, L_2, resp.

Construct NFA for $\mathrm{Merge}(L_1, L_2)$ as follows:

$A := \langle Q, \Sigma, \delta, q_0, F \rangle$

where

$Q = Q_1 \times Q_2$

$q_0 := (q_1, q_2)$

$F := F_1 \times F_2$

$\forall (q, r) \in Q, \ \forall a \in \Sigma,$

$\delta((q, r), a) = \{(\delta_1(q, a), r), (q, \delta_2(r, a))\}$

Def: Given lang L, def $\sqrt{L} := \{x : xx \in L\}$

Prop: If L is reg then $\sqrt{L}$ is reg.

Pumping Lemma: Let L be a regular language (over Σ). Then

(*) $\exists p > 0,$

$\forall s \in L$ such that $|s| \geq p$

$\exists x, y, z \in \Sigma^*$ such that

1) $s = xyz$
2) $|xy| \leq p$
3) $y \neq \varepsilon$,

$\forall i \geq 0, \ x\underline{y^i}z \in L$

If $i > 1$, this is "pumping up"

If $i = 0$, " " "pumping down"

If $i = 1$, this says nothing

(*) means that L is pumpable.

Pumping lemma (briefly):

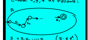
Every regular language is pumpable.

Proof: Let L be regular, recognized by a DFA A.

Let $p := \#$ of states of A.

Given $s \in L$, $|s| \geq p$,

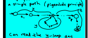
Choose x, y, z as follows:



$s = s_1 s_2 \cdots s_n \quad (s_i \in \Sigma)$

$n = |s| \geq p$

accept path of s cannot be a simple path (pigeonhole principle)



Can read the y-loop any # of times (incl 0), and still get to the accept state.

We have

$s = xyz$

$|xy| \leq p$

$y \neq \varepsilon$

and $\forall i \geq 0, \ x\underbrace{y^i}_{\text{go } i \text{ times around the } y\text{-loop}}z \in L$.

∎

Technique: use the pumping lemma to show that a language is not regular by showing that it is not pumpable.

Attachments

L5-8 - Preliminary Design (23.pdf

L5-8 - Preliminary Design (2).pptx