Deterministic Finite State Automata

Finite State Automata: TM without a tape

A deterministic finite state automate DFSA is (Σ, Q, δ, F) $\delta : Q \times \Sigma \to Q$ $F \subset Q$ are the accept states

The DFSA accepts if when it finishes reading the input, it is an accepting is state. Otherwise reject. ## Ex $L =$ binary numbers that are divisible by 8, noo leading zeros $\Sigma = \{0, 1\}$

q0: If 0, go to q1, else q3 q1: If end accept, otherwise go q2 (reject) q2: Stay q2, if end reject q3: Make sure ends with 3 0's on q6 to accept basically

Compliments

If L can be decided by DFSA A then $\bar{L}$ can be decided by a DFSA $\bar{A}$.

$\bar{A}$ has same Σ, δ, Q as A. But $F_{\bar{A}} = Q - F_A$

In other words, if we end on any state that would accept on A, we reject, otherwise accept.

Union

The union of 2 decidable by DFSA languages is also decidable by DFSA. Let L_1 be decimal numbers divisble by 3 Let L_2 be decimal number with even number of 0's Let $L_3 = L_1 \cup L_2$

L_3 states are $Q_3 = Q_1 \times Q_2$. That is the cross product of the states.

Regular languages

A language L is regular if it can be decided by a DFSA

Theorem

Regular langauges are closed under compliment, union, conccatenation, "star" operations

Proof later

Regular Expressions

- ϵ is a regular expression
- $a, a \in \Sigma$ is a regular expression
- $x \cup y$ where $x + y$ are regular expression
- $x \times y$ where x, y are regular expressions (concatenation)
- $x*$ (star operator is repeat 0 or more times) also reg ex

Note ϵ means move without reading the characters

Ex:

A reg ex for binary numbers devisable by by 8 (no leading 0's) $R = 0 + 1(0 + 1) * 000$ ## Theorem A language is regular iff it can be described by a regular expression

Proof

Given a REG EX, build a FSA

- $\epsilon : \delta(q_0, \epsilon) =$ accept
- $a : \delta(q_0, a) =$ accept
- $x \cup y$ assume you have a machine that accepts x and y, run in parallel and accept if one of them accepts
- xy assume you have a machine that accepts x and y, run sequentially and accept if both of them accepts
- $x*$ assume you have a machine that accepts x, we ϵ transition to accept or into our machine. If machine accepts we ϵ to accept. From accept we can ϵ to q_0

Given a FSA, build a REGEX

Repeateedly replace single states with "guarenteed" transitions.

Instead of using single characters for each transition we create REG EX for each transition.

Non Deterministic Finite State Automata

NFSA is (Σ, Q, δ, F)

$\delta : Q \times \Sigma \to P(Q)$

If there is a choice that will lead to the machine accepting the input the NFSA will make that choice otherwise it chooses at random.

Theorem

Given a NFSA N then there exists a determinstic FSA M that accepts the same language

Proof

$N = (\Sigma_N, Q_N, \delta_N, F_N)$

For M:

- $\Sigma_M = \Sigma_N$
- $q_{0M} = \{q_{0N}\}$
- $Q_M = P(Q_N)$
- $\delta_M(q_A, \sigma) = q_B$
  - $q_B = \{q_{yN} | \delta_N(q_x, x) = q_{yN} \quad \text{for} \quad q_x \in A\}$
- $F_M = \{q_A | \exists q_x \in F_N, q_x \in A\}$

Regular Languages

All finite langauges are regular because we can create a FSA that has all strings hardocded in.

If $s \in L, |s| > |Q|$ then some state will be repeated.

The Pumping Lemma

If L is a regular language, then there exists a positive integer p such that for all strings in L with $|s| > p$ then s can be divided into $s = xyz$ such that:

- $|xy| \leq p$
- $|y| > 0$

Then $xy^k z \in L$ for all $k \in \mathbb{N}$

In other words, we know the string length is larger than the number of states so we have a looping state. So the prefix (x) + loop (y) must be less than or equal to number of states $(z \geq 0)$. Additionally the looping portion must be non empty $(y > 0)$.

EX

Show $L = \{a^n b^m c^{n+m}\}$ is not regular by contradicting the pumping lemma

Proof

Assume L is regular, there exists a p.

Let $s = a^p b c^{p+1} = xyz$

xy is all a's

What is the string $xy^2 z$? -> $a^{p+|y|} b c^{p+1}$ If $xy^2 z \in L$ then $p + |y| + 1 = p + 1 \to |y| = 0$ contradiction!