CSE 596 Lecture Wed 8/28 Fall 2019

A deterministic finite automaton (DFA) is a 5-tuple $M = (Q, \Sigma, \delta, s, F)$ where:

Q is a finite set of states

Σ is a finite alphabet

s, a member of Q, is the start state

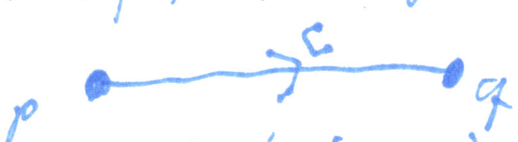
F, a subset of Q, is the set of {accepting states / desired final},

and δ is a function from $Q \times \Sigma$ to Q.

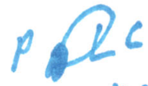
$\delta : Q \times \Sigma \to Q$

$\delta(p, c) = q$

Example Value: $p = q$ allowed.



instruction tuple (p, c, q)



loop arc on c

The rest of the lecture was a demo of the "Turing Kit" software for a DFA (and TM)

Type State can be int or anything else.

class DFA {

  set<State> Q;

  set<char> Sigma;

  State s; // start state

  set<State> F; // final states

  ~~State delta(State p, char c);~~

  State (*delta)(State p, char c);

  function pointer with instance-given code

  fine, but more general is:

  set<tuple> delta, where

  tuple = pair<pair<State, char>, State>

};

This defines a nondeterministic finite automaton (NFA). It is a DFA when the set δ of tuples has the property that for all $p \in Q$ and $c \in \Sigma$, there is exactly one $q \in Q$ s.t. $(p, c, q) \in \delta$.