Course 3
Finite Automata/Finite State Machines

The structure and the content of the lecture is based on (1) http://www.eecs.wsu.edu/~ananth/CptS317/Lectures/index.htm, (2) W. Schreiner Computability and Complexity, Lecture Notes, RISC-JKU, Austria

Excursion: Previous lecture

The Chomsky Hierarchy

We have: $\mathcal{L}_0 \supseteq \mathcal{L}_1 \supseteq \mathcal{L}_2 \supseteq \mathcal{L}_3$.

Closure properties of Chomsky families

Let $G_1 = (N_1, T_1, S_1, P_1)$, $G_2 = (N_2, T_2, S_2, P_2)$.

Closure of Chomsky families under union

The families $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ are closed under union.

Key idea in the proof

$$G_{\cup} = \left(V_{N_1 \cup N_2 \cup S}, V_{T_1 \cup T_2}, P_1 \cup P_2 \cup \{S \to S_1 | S_2\}\right)$$

Closure of Chomsky families under product

The families $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ are closed under product.

Key ideas in the proof

For $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2$

$$G_p = \left(V_{N_1 \cup N_2 \cup S}, V_{T_1 \cup T_2}, P_1 \cup P_2 \cup \{S \to S_1 S_2\}\right)$$

For $\mathcal{L}_3$

$$G_p = \left(V_{N_1 \cup N_2}, V_{T_1 \cup T_2}, S_1, P_1{}' \cup P_2\right)$$

where $P_1{}'$ is obtained from P_1 by replacing the rules $A \to p$ with $A \to pS_2$

Closure properties of Chomsky families (cont'd)

Closure of Chomsky families under Kleene closure

The families $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ are closed under Kleene closure operation.

Key ideas in the proof

For $\mathcal{L}_0, \mathcal{L}_1$

$$G^* = (V_N \cup \{S^*, X\}, V_T, S^*, P \cup \{S^* \to \lambda | S | XS, Xi \to Si | XSi, \quad i \in V_T\})$$

The new introduced rules are of type 1, so G^* does not modify the type of G.

For $\mathcal{L}_2$

$$G^* = (V_N \cup \{S^*\}, V_T, S^*, P \cup \{S^* \to S^*S | \lambda\})$$

For $\mathcal{L}_3$

$$G^* = (V_N \cup \{S^*\}, V_T, S^*, P \cup P' \cup \{S^* \to S | \lambda\})$$

where P' is obtained with category II rules, from P, namely if $A \to p \in P$ then $A \to pS \in P$.

Finite Automata

Finite Automaton (FA)

Finite state machines are everywhere!

https://www.youtube.com/watch?v=t8YKCItVDlg

Why finite automata are important?

https://www.quora.com/Why-is-it-so-important-to-have-a-good-understanding-of-automata-theory

Finite Automaton (FA)

- Informally, a state diagram that comprehensively captures all possible states and transitions that a machine can take while responding to a stream or sequence of input symbols.
- Recognizer for “Regular Languages”

- Deterministic Finite Automata (DFA)
  - The machine can exist in only one state at any given time
- Non-deterministic Finite Automata (NFA)
  - The machine can exist in multiple states at the same time

Deterministic Finite Automata - Definition

- A Deterministic Finite Automaton (DFA) consists of:
  - Q - a finite set of states
  - Σ - a finite set of input symbols (alphabet)
  - q_0 - a start state (one of the elements from Q)
  - F - set of accepting states
  - $\delta : Q \times \Sigma \rightarrow Q$ - a transition function which takes a state and an input symbol as an argument and returns a state.
- A DFA is defined by the 5-tuple: $\{Q, \Sigma, q_0, F, \delta\}$

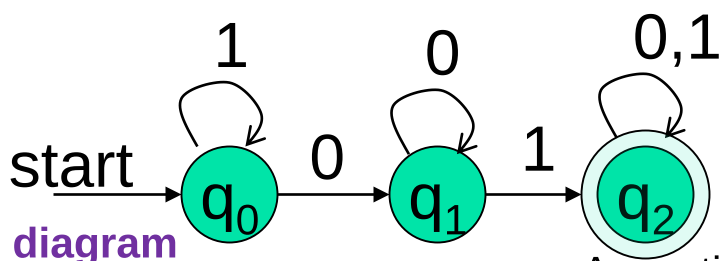
Example #1

- Build a DFA for the following language:
  - L = {w | w is a binary string that contains 01 as a substring} same as
  - L = {w | w is of the form x01y where x,y are binary strings} same as
  - L = {x01y | x,y are binary strings}
  - *Examples*: 01, 010, 011, 0011, etc.
  - *Counterexamples*: ε, 0, 1, 111000
- Steps for building a DFA to recognize L:
  - $\sum = \{0,1\}$
  - Decide on the non-final (non-accepting) states: Q
  - Designate start state and final (accepting) state(s): F
  - Decide on the transitions: δ

Regular expression: (01)*01(01)*

DFA for strings containing 01

- What makes this DFA deterministic?



Transition diagram

- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{0,1\}$
- start state = q_0
- $F = \{q_2\}$

Transition table

symbols

δ	0	1
→ q_0	q_1	q_0
q_1	q_1	q_2
*q_2	q_2	q_2

Start state

states

Accepting/final state

What if the language allows empty strings?

Summary

- Finita Automata
  - Deterministic
  - Non-deterministic