Example3

- {x01y: x, y are any strings of 0’s and 1’s}

Extention

- Recall δ
- Extend it with δ head
- Instead of symbol, use string as input to δ head

- Form of an Inductive Definition
  - Base case: for **Input** 0, 1, or empty string ε
  - Iductive part: for **any Given** input

Inductive Defs

- Form:
  - $f(0, y) = g(y)$
  - $f(n+1, y) = h(n, y, f(n,y))$
- Example1:
  - $y^0 = 1$
  - $y^{n+1} = y^n . y$ $\quad$ $g(y) =1, h(x,y,z) = z . y$
- Example2:
  - Cumulative sum of c_i $(i=0, \dots n)$
  - …
  - $G(y) = c_0$, $h(x,y,z) = z + c_{x+1}$

Inductive defs (cont'd)

- Boolean function:
  - Inputs:x_1, ... x_n;   outputs: y_1, ... y_m
  - Interested in case when m=1
- Such a boolean fcn can be represented as a finite table of 2^n entries
  - Or a boolean expression, consisting of
    - Boolean vars,
    - Boolean constants
    - Boolean operations (and, or, not)

Boolean Function

- The Cartesian product:

$X_1 \times X_2 \times \ldots \times X_n = \{(x_1, x_1, \ldots, x_n): x_1 \varepsilon X_1, \ldots, x_n \varepsilon X_n\}$

$X^n = X_1 \times X_2 \times \ldots \times X_n$ when $X_1 = X_2 = \ldots = X_n$

- A boolean function f is any function $f: B^n \rightarrow B^m$
- Interested in case where m=1
- Repr1:
  - A finite table with 2^n entries
- Repr2:
  - A boolean expression consists of
  - Boolean vars
  - Boolean constants
  - Boolean operators

Boolean Expr (inductive defs)

- Any boolean var is a boolean expression, and any constant is a boolean expression.
- If e_1 and e_2 are boolean exprs, then so are (e_1 **or** e_2), (e_1 **and** e_2) and (**not** e_1)
- Every boolean expr with n vars represents some boolean function f: $B^n \rightarrow B$
- Theorem: Every boolean function f: $B^n \rightarrow B$ is represented by some boolean expr with n vars.

Standard Notation

- Small letters a, b, c and subscripted ones at the beginning of alphabet refer to just a symbol (elements of Σ)
- Small letters x, w, y, z and subscripted ones at the end of the alphabet refer to a string over Σ

Extention

- Recall δ
  - $\delta : Q \times \Sigma \rightarrow Q$
- Extend it with δ head
- Instead of symbol, use string as input to δ head

$$\delta \text{ head} : Q \times \Sigma^* \rightarrow Q$$

$$\delta \text{ head} (q_0, x)$$

- Inductive definition for δ head
  - Base: no input, ε
  - Iductive part: string w =xa; a is last symbol of w.

Language accepted by FSA

- Given any $M = (Q, \Sigma, \delta, q_0, F)$
- $L(M)$ = set of strings accepted by M
- $L(M) = \{x \text{ in } \Sigma^* : \delta \text{ head } (q_0, x) \text{ in } F\}$

Ultimately

- Given a language L_1
- Find FSA $M=(Q, \Sigma, \delta, q_0, F)$
  - such that $L(M) = L_1$

Finite State Automata(FSAs)

- Deterministic FSA (DFAs)
- To define the language of a DFA A = $(Q, \Sigma, \delta, q_0, F)$
- L(A) = {w : δ head (q0, w) is in F}
- Set of strings w that take A from the start state q0 to one of accepting states.
- A lang L1 is regular if there is a DFA M such that L(M)= L1
- If a given L is L(A) for some DFA A, then we say L is a regular language.

Example4

- L= {x: 0110 is a substring of x}
- Is L regular?

Sequences

- Two sequences
- Input: sequence of symbols
- State that M is in. Sequence of states
- Number of states = Number of inputs + 1

A lang is regular

- Given A= {x: 0110 is a substring of x}
- Is A regular?
- One way: find a DFA M s t L(M) = A

Complement of L

L1 U L2

$L1 \cap L2$