Transition Diagrams

Tuesday, March 1, 2022 3:31 PM

3.4.1 Recognition of Tokens: Transition Diagrams

↳ Transition diagrams have a collection of nodes/circles, called STATES
→ Each STATE represents a condition that could occur scans until a pattern is found

STATE → all characters between the lexeme Begin and forward pointer

↳ EDGES are directed from one state to another.
→ Each EDGE is labeled by a symbol / set of symbols
i.e. If we're in state S and the next input symbol is "a",
we look for the edge out of state S labeled "a".
↳ if we find it, we advance the forward pointer and enter the STATE were EDGE labeled "a" ends.

→ DETERMINISTIC: a diagram with never more than ONE EDGE out of a STATE with a given symbol in its labels.

1. Certain states are said to be *accepting*, or *final*. These states indicate that a lexeme has been found, although the actual lexeme may not consist of all positions between the *lexemeBegin* and *forward* pointers. We always indicate an accepting state by a double circle, and if there is an action to be taken — typically returning a token and an attribute value to the parser — we shall attach that action to the accepting state.
2. In addition, if it is necessary to retract the *forward* pointer one position (i.e., the lexeme does not include the symbol that got us to the accepting state), then we shall additionally place a * near that accepting state. In our example, it is never necessary to retract *forward* by more than one position, but if it were, we could attach any number of *'s to the accepting state.
3. One state is designated the *start state*, or *initial state*; it is indicated by an edge, labeled "start," entering from nowhere. The transition diagram always begins in the start state before any input symbols have been read.

* Computational Grammar (Karen's Notes)

→ alphabet: set of symbols
- finite
- non-empty
- usually represented with "Σ" sigma

i.e.: english alphabet
$$\Sigma = \{a, b, c, d, e, \ldots, z\}$$

i.e.: binary alphabet
$$\Sigma = \{0, 1\}$$

→ string: ordered sequence of symbols from the alphabet with an specified length (num of characters)
i.e.: 1101 is a string of alphabet $\Sigma = \{0,1\}$ with length(4)
- Empty string: length $\emptyset$, represented by epsilon "ε".

→ Power of an alphabet: set of all strings of a certain length that can be formed with an alphabet.

i.e. $\Sigma = \{0,1\}$ → alphabet
$\Sigma^1 = \{0,1\}$ → power 1 set
$\Sigma^2 = \{00, 01, 10, 11\}$ → power 2 set:
$\Sigma^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$ → power 3 set:
$\Sigma^0 = \{\varepsilon\}$

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \ldots$
Sigma star: all possible combinations (union of all power sigmas)

$\Sigma^+ = \Sigma^* - \Sigma^0$

Sigma plus: all but the empty string

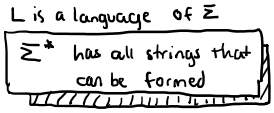
→ Concatenation

x = "hello" y = "world" xy = "helloworld"

$x\epsilon = \epsilon x = x$

→ LANGUAGE: a language is a subset of sigma Star

$L \subseteq \Sigma^*$ L is a language of Σ



* The Problem: to decide if a string is part of a language

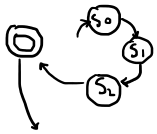
↳ all programs are a problem → computer program } AUTOMATA

↓

YES / NO → series of steps to decide if a string is part of a language

FINITE AUTOMATA

buffer: | 1 | ø | 1 |



→ states that depend on what is read in a buffer, they change to a different state based on this.

→ If we manage to reach the Acceptance State, the string belongs/is part of the language

FINITE AUTOMATA ~ machine that solves a problem.

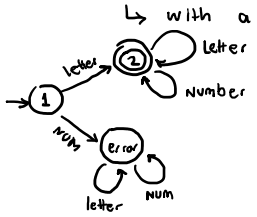
i.e. generate an automata to determine if an identifier is valid or not.

→ begins with a letter

→ continues with letters or numbers

¿HOW DO WE REPRESENT AN AUTOMATA?

↳ with a Transition Diagram



ELEMENTS:

→ Transition: what does the machine do if it receives a certain symbol?

○ State: takes different decisions depending on its transitions

◎ Acceptance state: if we finish the reading here, the string is valid.

→① Initial state: by default we begin here

FINITE AUTOMATA(A): It is a tuple formed by:

$A = \{ø, \Sigma, \partial, i, F\}$

ø: finite set of states

Σ: alphabet

∂: Transition function

$\partial: ø \times \Sigma \to ø$

i: initial state $i \subseteq ø$

F: set of final states $F \subseteq ø$

lenguaje : $\{w \mid$ w contiene la subcadena 01 $\}$

$Q = \{q_0, q_1, q_2\}$
$\Sigma = \{0, 1\}$
$i = q_0$
$F = \{q_1\}$

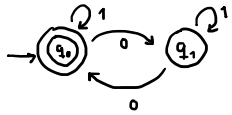
δ : para cada estado y cada elemento del alfabeto

δ (Símbolo / estado)	0	1
→ q_0	q_2	q_0
* q_1	q_1, *	q_1, *
q_2	q_2	q_1, *

→ (inicial)
* (aceptación)

i.e. Generate the diagram of an automata that accepts the language:

a) $L_1 = \{w \mid w$ has an even number of ceros $\}$



Zero zeroes is also even.