CS 302 Automata Theory Spring 2021

Text :

Introduction to Automata Theory, Languages and Computation

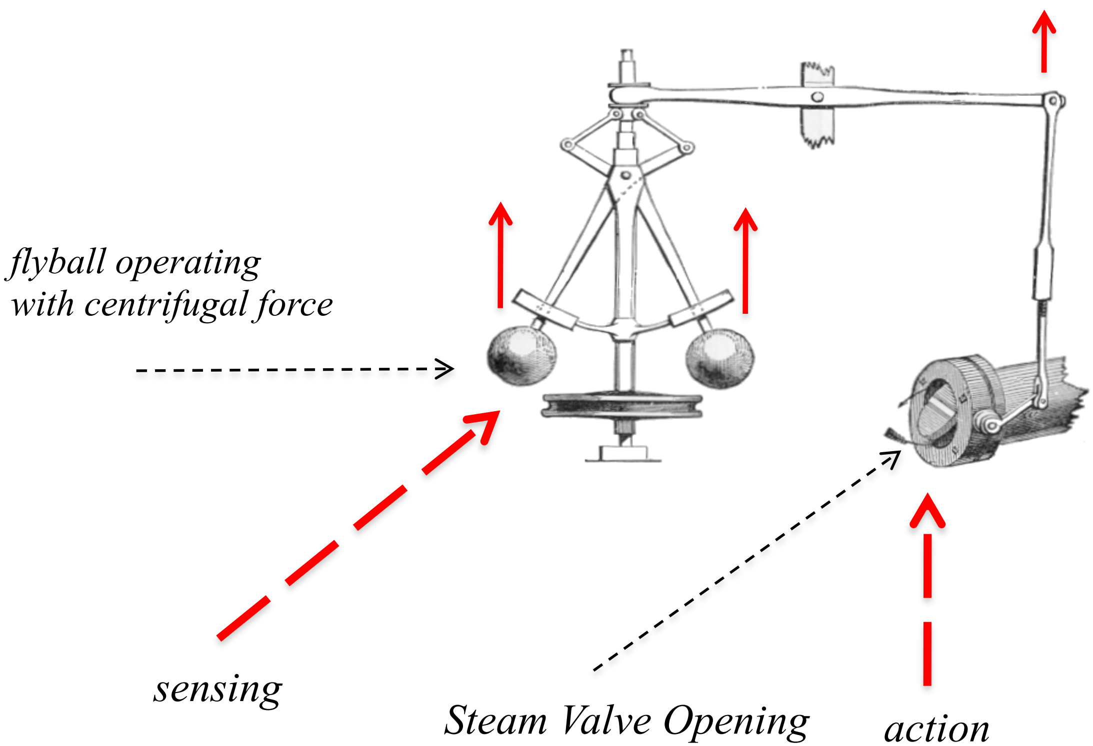
Third edition, Pearson 2006

Instructor :

Kemal İnan, inan@sabanciuniv.edu

PREMODERN AUTOMATION

James Watt's Governor (Speed regulator) 1788

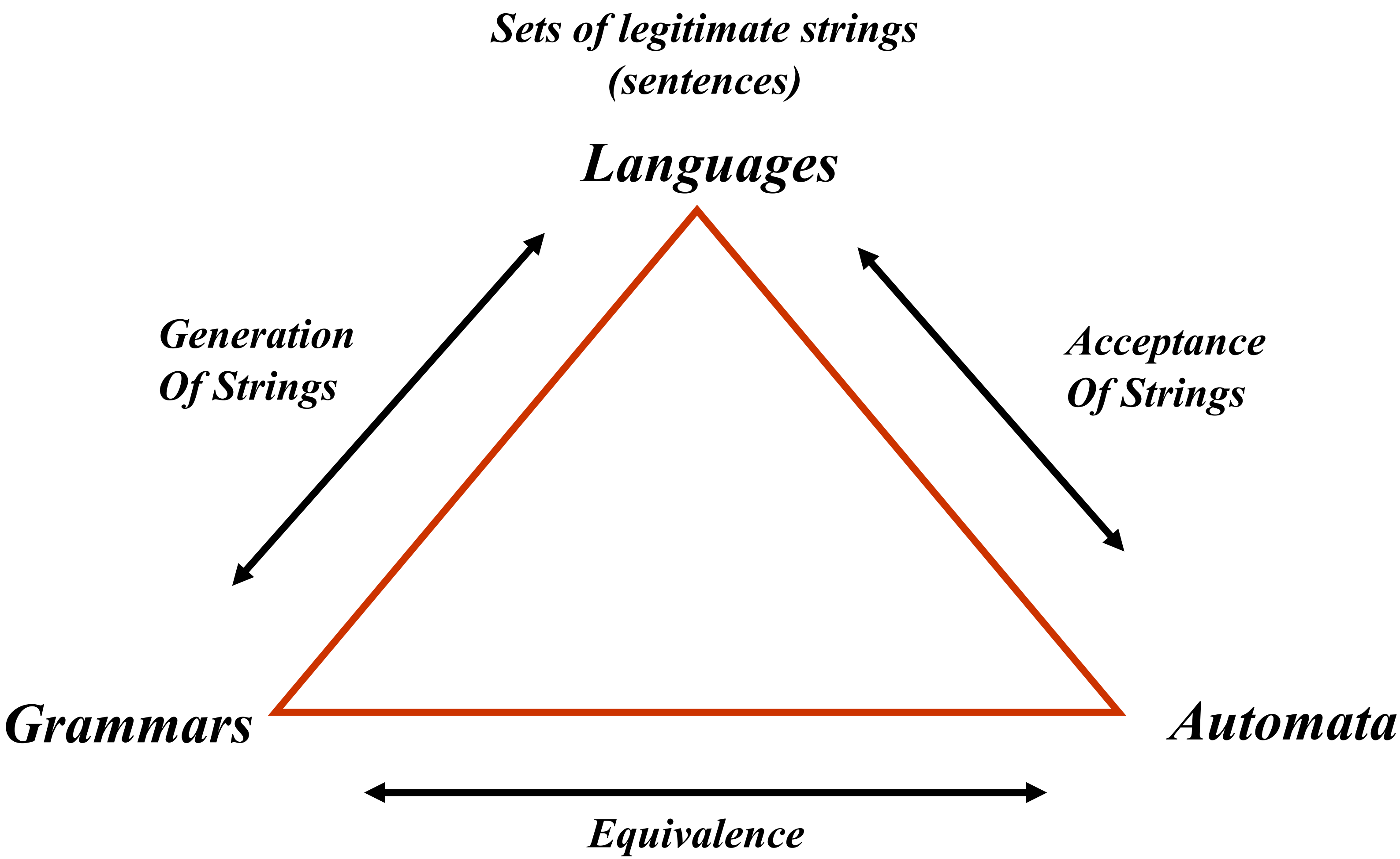


Almost a century later : ***James Clerk Maxwell's famous paper (1868) : On Governors***

First mathematical treatment of stability

Classical Example of Negative Feedback

Topic of the Course → MODERN : LINGUISTIC BASED AUTOMATION



Definition of a Language

(1) A finite set $\boldsymbol{\Sigma}$, *called the* ***alphabet*** *set.*

(2) A ***set*** *of strings with elements in* $\boldsymbol{\Sigma}$ *is called a* ***language*** *over* $\boldsymbol{\Sigma}$

Formal Definition of a Language $L \subseteq \Sigma^*$ where :

emptystring

$$\Sigma^* := e \cup \Sigma^+$$

$$\Sigma^+ := \Sigma \cup \Sigma^2 \cup \ldots \cup \Sigma^n \cup \ldots = \cup_{k=1,+\infty} \Sigma^k$$

where *n times*

$$\Sigma^n := \Sigma . \Sigma . \ldots . \Sigma := \{\sigma_1 \sigma_2 \ldots \sigma_n \mid \sigma_j \in \Sigma, j=1,2,\ldots,n\}$$

= *set of all strings with elements in* Σ *of length* $\boldsymbol{n}$

String Operations and Terminology

String Concatenation : $u \in \Sigma^*, v \in \Sigma^* : u.v \in \Sigma^*$

$u \in \Sigma^+, v \in \Sigma^+$ *and* $w \in \Sigma^+$; $\mathbf{s} = \boldsymbol{u} \,.\, \boldsymbol{v} \,.\, \boldsymbol{w}$;

each ***u*****,*****v*** *or* ***w*** *is called a substring of s ;*

u *is called a prefix of s*

w *is called a postfix of s*

$\boldsymbol{s}^{\boldsymbol{n}}$ *denotes a string* ***s*** *concatenated with itself* ***n*** *times*

length (***s***) = # *characters in* ***s*** = |***s***|

*How can we define a language **L** ?*

$$L := (s \in \Sigma^* \mid F(s))$$

*A logical condition on **s** ; **F** is a truth valued function*

There is a problem in this definition :

*Is **F** computable ?*

*What does **computable** mean ?*

*Two **computable** tools are introduced :*

(1) Grammars ; (2) Automata

Examples of languages :

***(1) Natural Languages** ; strings of characters from a keyboard that are **syntactically** correct in **English** e.g. **The chair ate the elephant** is a syntactically correct string (sentence) in the English language ; **The ate elephant chair the** is not correct !*

***(2) Formal** (Computer**) Languages :** e.g. strings of symbols that are syntactically correct ; such as a C++ program for which the compiler does not give a syntax error*

Simple examples of formal languages

(3) *Well-defined expressions. eg. arithmetic expressions using the operators* + *and* × *and nonnegative integers*

Operation not specified

$(32+560)\times(3+54\times 7)$ *is correct whereas* $32\,(+056\times 7\,(3))$ *is not correct*

integer cannot start with a 0

(4) *Problems :* ***encoding ?*** *of* ***decision ?*** *problems*

Examples :

(i) Decision problem :

$$E = \{\, (n, m, k) \in Z \times Z \times Z \mid n+m = k\ ?\, \}\ ;\ E \subseteq Z^3$$

(ii) Encoding of a graph **G** *that solves the decision problem of connectedness !*

Context Free Grammars

$$G = (\; V \;,\; T \;,\; P \;,\; S \;)$$

Terminal symbols $T \subseteq V$

Start symbol $S \in V\text{-}T$

Finite set of symbols

Finite set of production rules
$P \subseteq (V\text{-}T) \times V^*$

Example : generation of integers in decimal notation

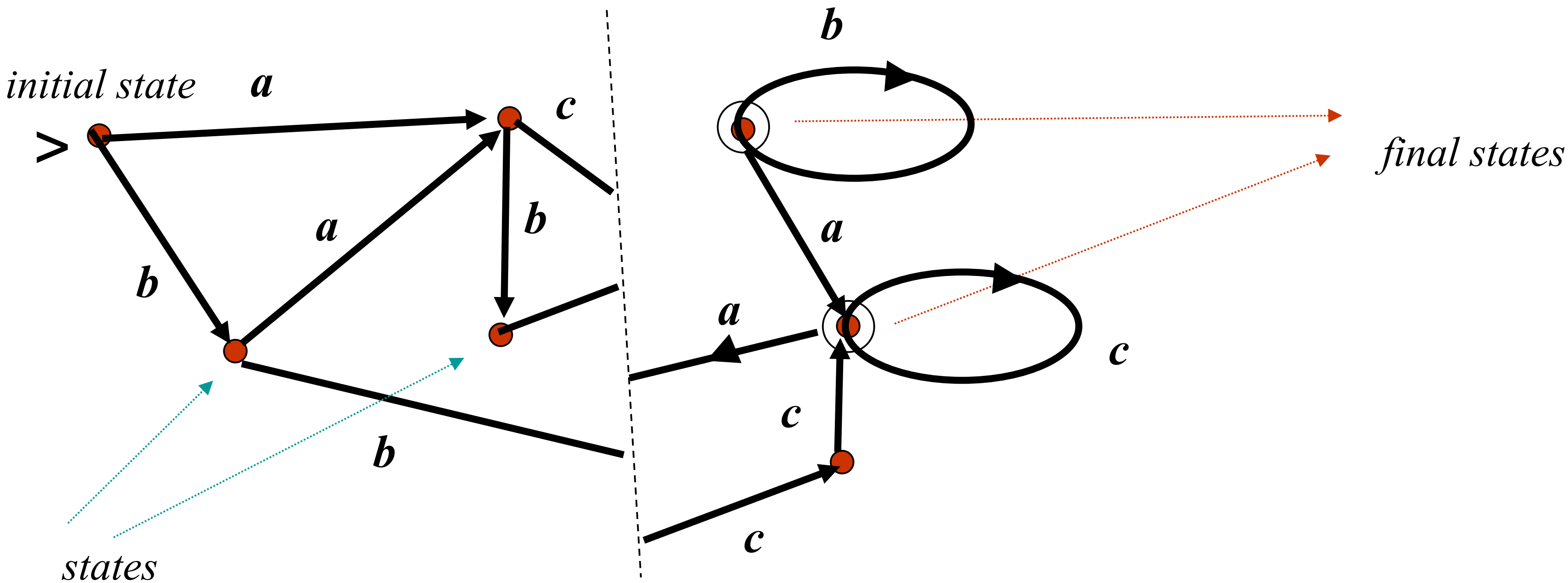
-108970 and +67 and 564 are legitimate strings but 034

and 1-3 and 90+1 are not!

Find a grammar that generates integers in decimal notation !

(Deterministic Finite) Automata over a set $\Sigma = \{a, b, c, \ldots\}$

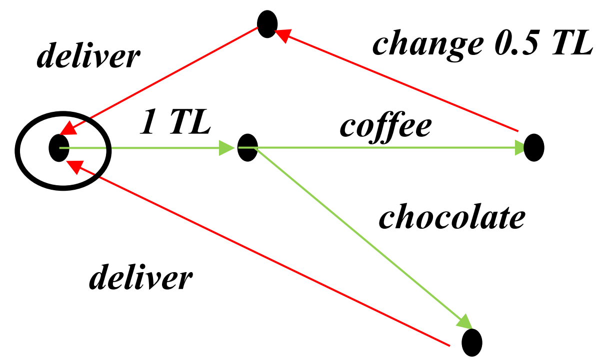
Simple way to define is by directed graphs where edges are labeled by symbols in Σ



In ***CS 302*** *we use* ***Automata*** *as a language acceptor (or generator)*

But there are other uses in modeling real systems :

(1) Coffee & Chocolate Machine



(2) Digital Integrated Circuits with input and output