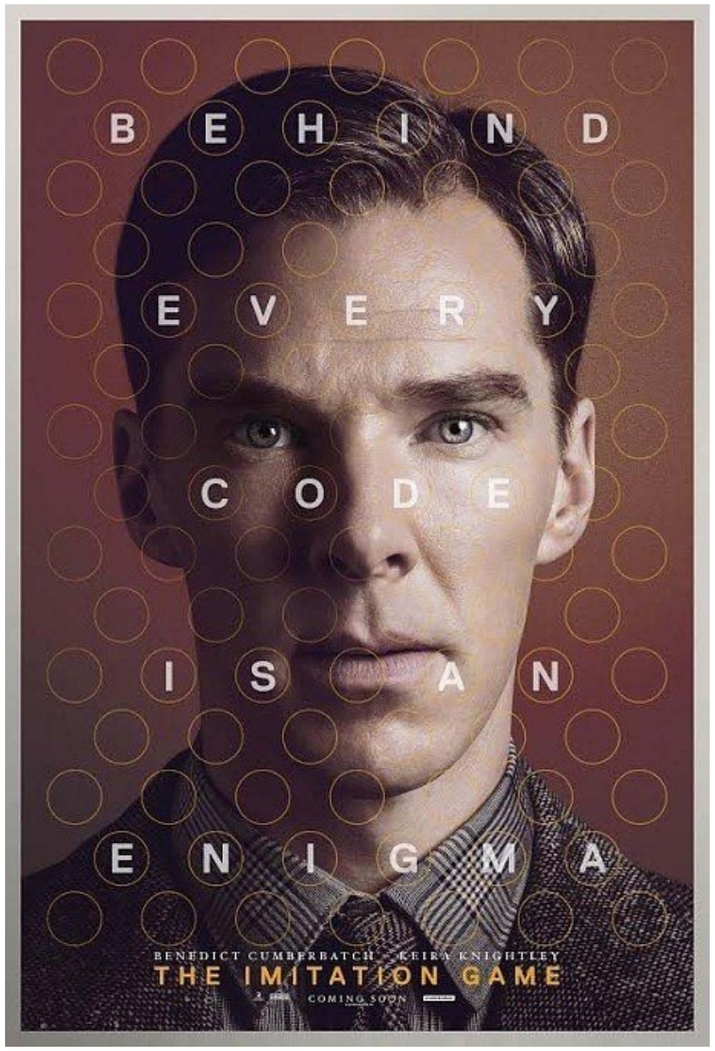
Automata & languages

A primer on the Theory of Computation



Laurent Vanbever

nsg.ethz.ch

ETH Zürich (D-ITET)

15 October 2026

Part 5

Last week was all about

Context-Free Languages

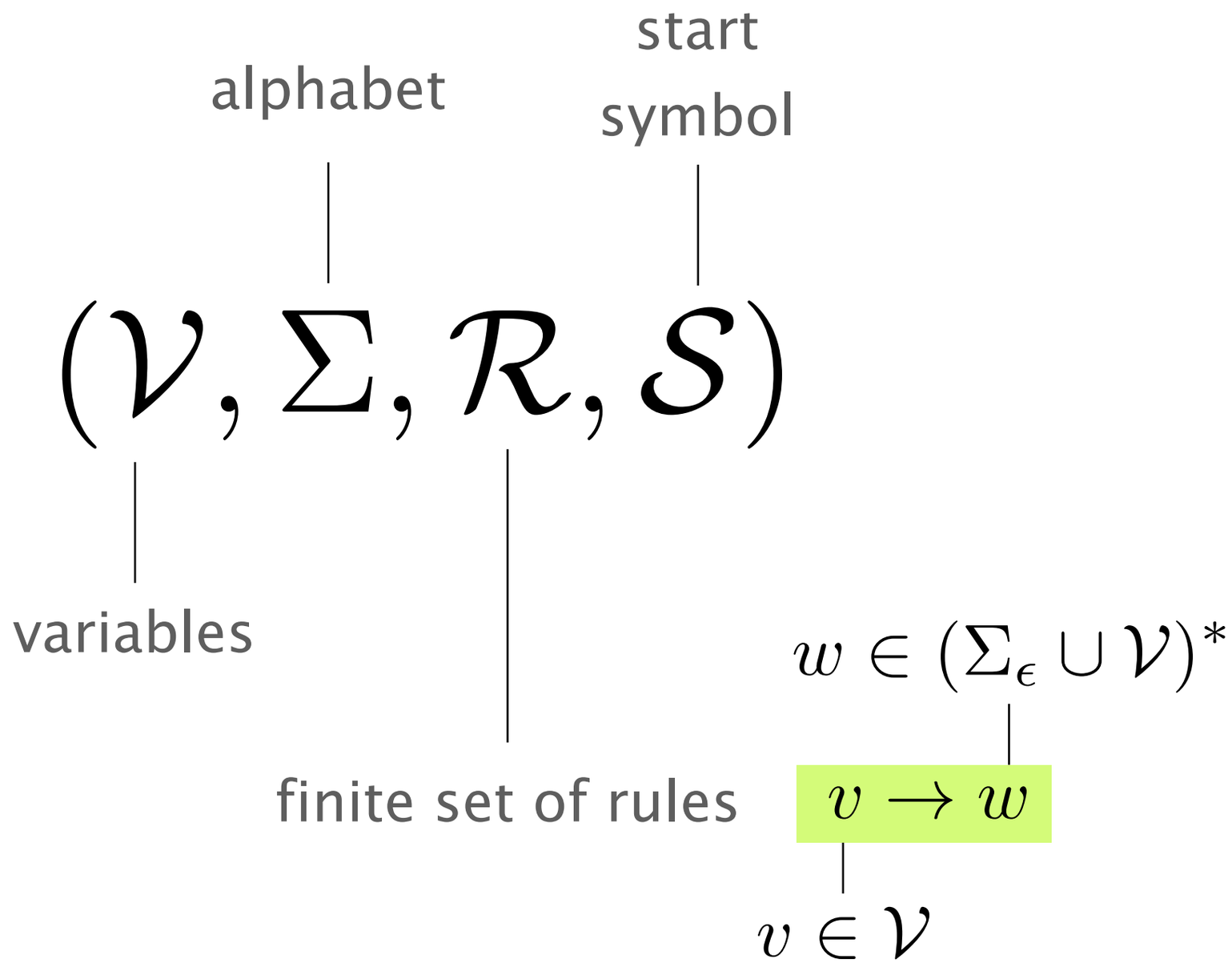
Context-Free Languages

a superset of Regular Languages

Example $\{0^n1^n \mid n \geq 0\}$ is a CFL but not a RL

We explored the concepts behind

Context-Free Grammars



As for Regular Languages,

Context-Free Languages are recognized by “machines”

Language	Regular	Context-Free
Machine	DFA/NFA	Push Down Automata

Push-Down Automatas are pretty similar to DFAs

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

alphabet

start state

states

accepting states

Push-Down Automatas are pretty similar to DFAs except for… the stack and transition function

stack alphabet

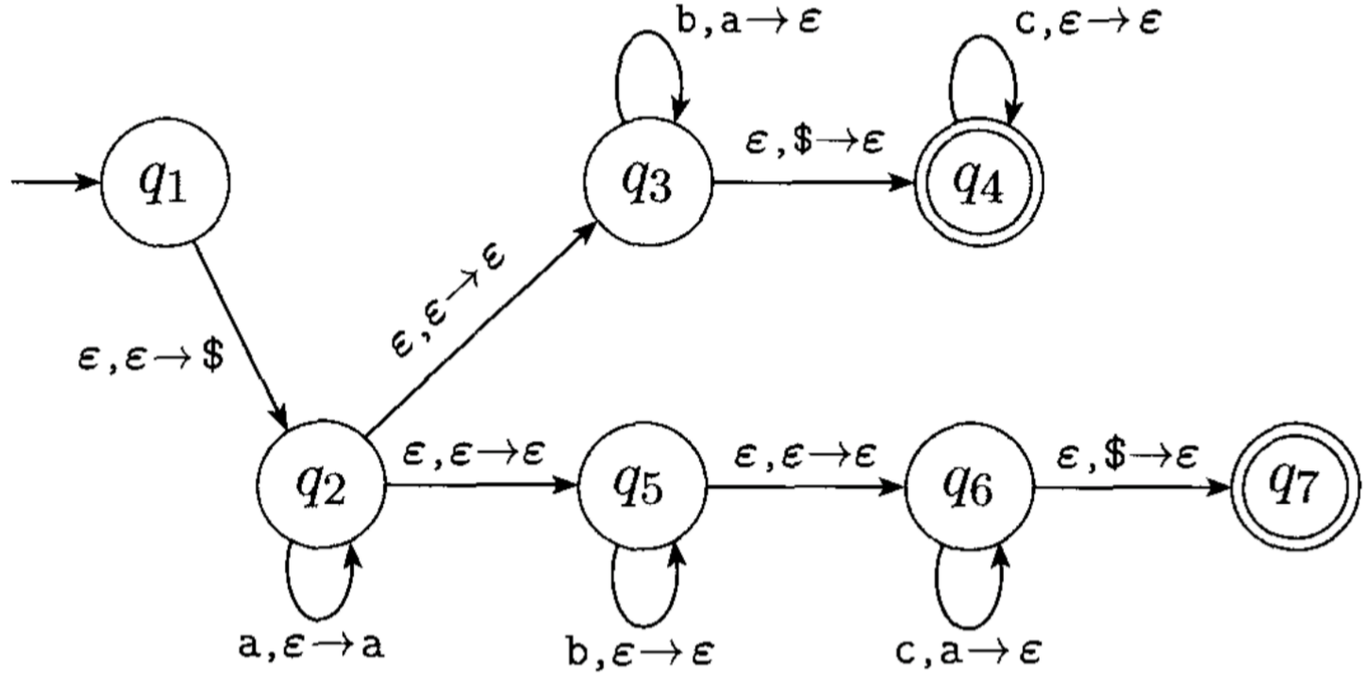
$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

transition function

$$Q \times \Sigma_\epsilon \times \Gamma_\epsilon \to P(Q \times \Gamma_\epsilon)$$

Language $L = \{a^i b^j c^k \mid i,j,k \geq 0 \text{ and } i=j \text{ or } i=k\}$

Machine (PDA)



This week, we'll confirm that

computers are not limitless

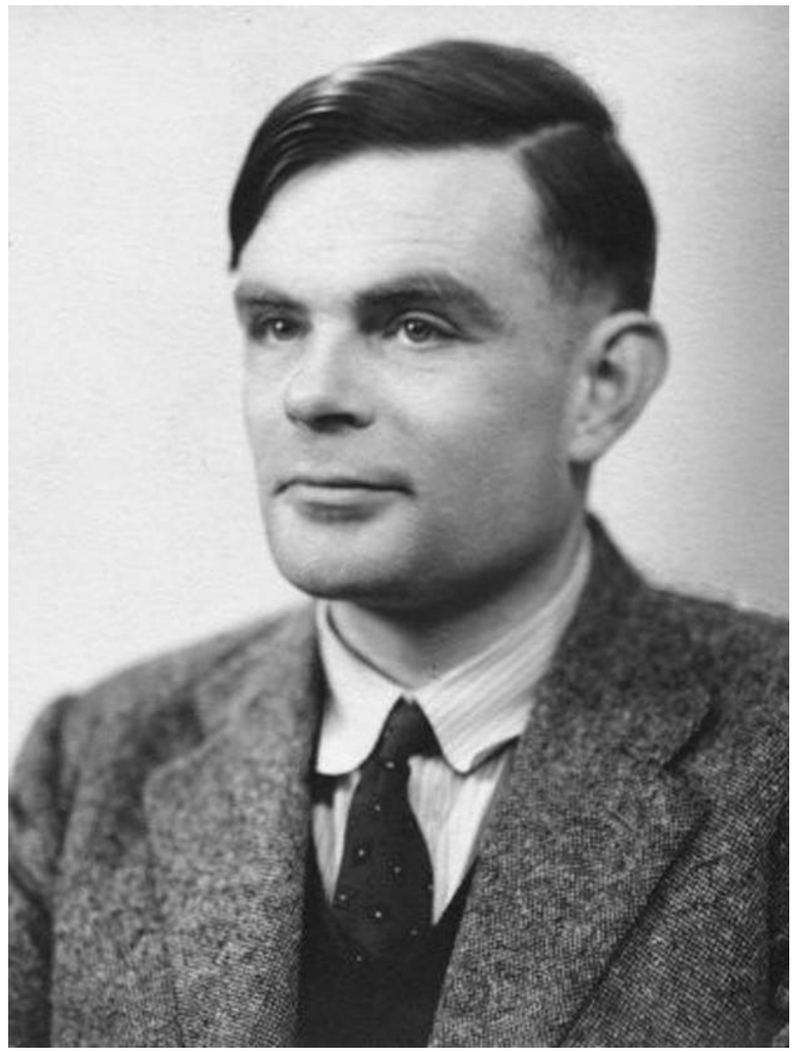
Some problems

cannot be solved

by a computer

(no matter its power)

Alan Turing (1912-1954)



This week we'll leave the confines of CFL
and introduce the concept of a Turing machine

Today's plan

Thu Oct 15

1. PDA ≃ CFG
2. Pumping lemma for CFL
3. Intro to Turing machines

In particular,

we'll see that not all languages are CFL

Language $L = \{ w\#w \mid w \text{ in } \{0,1\}^* \}$

This is *not* a CFL!

Is the stack in a PDA fundamental?

Could we use other data structures such as a queue?

Stack-based PDAs can handle nested structures

symbols are pushed and popped in reverse order

Queue-based PDAs cannot handled nested structures

they cannot reverse the order of the symbols

They recognize different languages!

a queue-based PDA can easily recognize { w#w }