# Representing Languages II

Discrete Mathematics

Evgeny Skvortsov

# Previous Lecture

- Kleene star
- Regular expressions

# Grammars

- What is a sentence in a natural language?
- One typical rule is: A sentence can consist of a noun phrase followed by a predicate.

  We may represent this rule as

  <sentence> → <noun_phrase> <predicate>

- This is not enough to deal with real sentences, and we need to explain <noun_phrase> and <predicate>

  <noun_phrase> → <article> <noun>

  <predicate> → <verb>

# Grammars (cntd)

- This is still not enough as we need concrete words in a sentence, so we continue

  <article> → the | a

  <noun> → dog | boy | …

  <verb> → walks | runs | …

- If we describe all possible constructions, we get a full description of possible sentences

# Grammars (example)

- Programming languages are described in terms of grammars

<conditional statement> ::= <if statement> | <if statement> **else** <statement> | <if clause> <for statement> | <label>: <conditional statement>

<if clause> ::= **if** <Boolean expression> **then**

<unconditional statement> ::= <basic statement> | <compound statement> | <block>

<if statement> ::= <if clause> <unconditional statement>

<Boolean expression> ::= <simple Boolean> | <if clause> <simple Boolean> **else** <Boolean expression>

<simple Boolean> ::= <implication> | <simple Boolean>

…

# Definition

- A grammar G consists of
  - V set of variables
  - $\Sigma$ set of terminals (or terminal symbols)
  - $S \in V$ a start symbol
  - P a set of rules or productions
- Every production has the form

  $x \rightarrow y$

  where x is a non-empty string consisting of terminals and variables, and y is any string consisting of terminals and variables

  <noun_phrase> runs → <article> dog runs

  <if clause> ::= if <Boolean expression> then

# Derivation

- If we have a rule $x \rightarrow y$, and a string of the form $w = uxv$, then we can use the rule to derive the string $z = uyv$

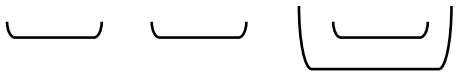
  $w = uxv \Rightarrow uyv = z$

  w derives z
- Rule $S \rightarrow aSb$

  $abSbaSbaSaba \Rightarrow abSbaSbaaSbaba$

  
- If $w_1 \Rightarrow w_2 \Rightarrow \ldots \Rightarrow w_n$ then we write $w_1 \Rightarrow^* w_n$
- Example

  $abSbaSbaSaba \Rightarrow^* abaSbbaaSbbaaaSbbaba$

# Generating a Language

- Let $G = (V,\Sigma,S,P)$ be a grammar. Then the language $L(G)$ generated by $G$ is the set of all strings of terminal symbols that can be derived from $S$

  Formally,

  $$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

- What is the language generated by the following grammar?

  $S \rightarrow Aa$ $\quad A \rightarrow \lambda$

  $A \rightarrow Ba$ $\quad B \rightarrow \lambda$

  $A \rightarrow Bb$ $\quad B \rightarrow abba$

# More Examples

- What is the language generated by the following grammar?
  $S \rightarrow aSb \qquad S \rightarrow \lambda$

- What is the language generated by the following grammar?
  $S \rightarrow SS \qquad S \rightarrow \lambda \qquad S \rightarrow aSb \qquad S \rightarrow bSa$

- Suggest a grammar that generates the language of properly placed parenthesis.

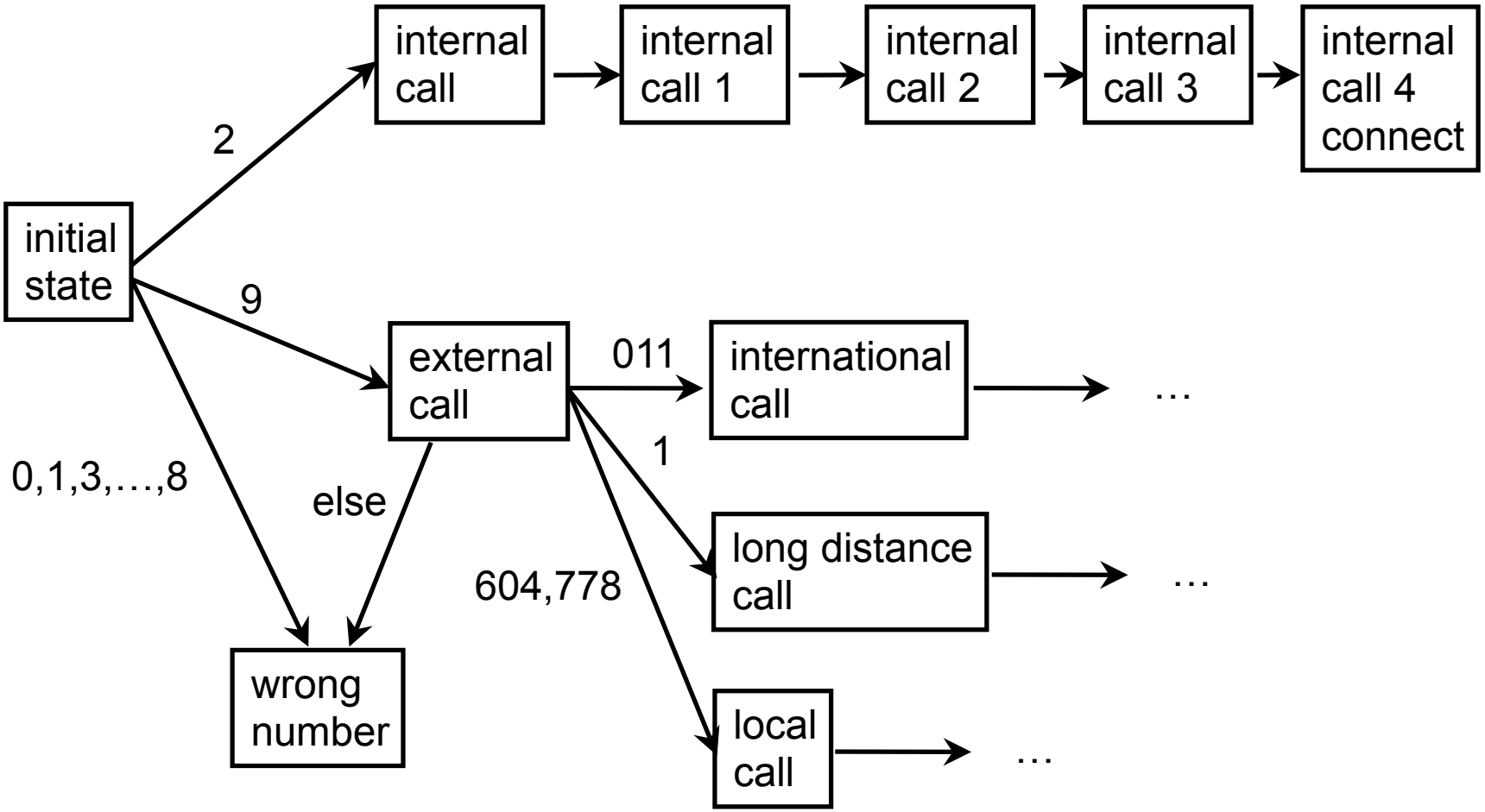
# Finite Automata: Example

- Consider how a telephone station works when we dial a number.
- If the first dialed digit is 9, it is an external call
  If the first digit is 2, it is an internal call
  Otherwise it is a wrong number
- In the case we are making an internal call, the number dialed consists of four digits, not including the first digit 2
  If the dialed number has different length, it is a wrong number
- In the case we are making an external call
  If the first three digits are 011, it is an international call and the following number should consists of a valid country and area code, and a number of proper length
  If the first digit is 1, it is a long distance call
  If the first three digits are 604 or 778, it is a local call, and the following number should contain seven digits
  Otherwise it is a wrong number

# Finite Automata: States and Transitions

- The logic of the telephone station can be described in terms of internal states
- When we lift the receiver, the station switches to the initial state
- Every time we dial a digit, the station switches to another state depending on the digit dialed, and possibly gives some output (for example, `The number you have dialed is not recognizable')
- When we stop dialing, depending on the current state, the station connects to the required number or tells that the number is wrong
- Every change of an internal state is called a transition

# Finite Automata: States and Transitions



# Finite Automata: Definition

- A finite automaton consists of:
  - a set of internal states $S$
  - the input alphabet $\Sigma$
  - the transition function $v$
  - the initial state $s_0$
  - the set of accepting states $F \subseteq S$
- The transition function is a collection of rules of the form

  $$(s,a) \rightarrow s'$$

  where $s$ is the current state of the automaton, $a \in \Sigma$ is an input symbol, and $s'$ is the new state

  The transition function contains such a rule for every pair from $S \times \Sigma$

# Finite Automata: Definition (cntd)

- Input of the automaton is a string w
- The automaton starts in the initial state
- The automaton reads w from left to right
- After reading each symbol it changes its state accordingly to the transition function
- It stops after reaching the end of w

- Example: $S = \{s_0, s_1\}$, $\Sigma = \{a,b\}$, $s_0$ is the initial state

  $v = \{ (s_0,a) \to s_1, \quad (s_0,b) \to s_0,$
  $(s_1,a) \to s_0, \quad (s_1,b) \to s_1 \}$

  Input: aabaa

# Recognizing Languages

- A finite automaton accepts a string w, if its work with input w ends up in an accepting state
- Example (see previous slide): $F = \{ s_0 \}$

  Inputs: aabaa, abbaba

- Let M be a finite automaton

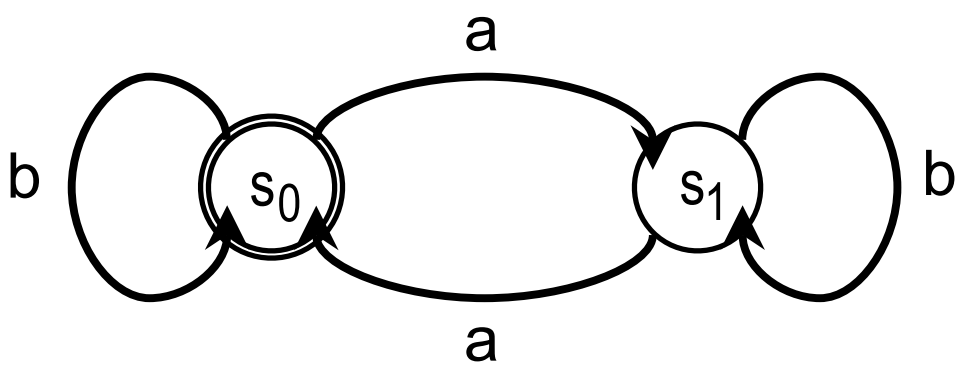
  $L(M) = \{w \mid M \text{ accepts } w\}$

  is the language accepted by M

- What is the language accepted by the automaton from the previous slide?

# Graph of a Finite Automaton

- It is convenient to represent finite automata as directed graphs
- States correspond to vertices of the graph
- Every rule $(s,a) \rightarrow s'$ of the transition function is represented by an arc (or edge) from $s$ to $s'$ labeled by the symbol $a$
- The initial state is always $s_0$
- Final states we encircle twice



# Examples

- Draw the graph of the automaton:

  $S = \{s_0, s_1, s_2, s_3\}$, $\Sigma = \{a,b\}$, $F = \{ s_1 \}$

  $v = \{ (s_0,a) \to s_1,\ (s_0,b) \to s_3,$
  $(s_1,a) \to s_2,\ (s_1,b) \to s_3,$
  $(s_2,a) \to s_3,\ (s_2,b) \to s_1,$
  $(s_3,a) \to s_3,\ (s_3,b) \to s_3 \}$

  What language does it accept?

- Construct a finite automaton that accepts the language $a^* ba^* ba^*$

# Homework

Exercises from the Book:

No. 3, 4, 6 (page 332)

Suggest a grammar that generates correct algebraic expressions