

# FOUNDATIONS OF MATHEMATICS, LECTURE 6

András Kornai

BMETE91AM35 Fall 2021-22

# LOGIC

- The twin pillars of the foundation are set theory and logic
- We already started to build relations and functions on set theory, and we will continue with operations and structures. But now we turn to logic
- Set theory has variants (ZFC, NGB, KP . . . ), logic has many more variants!
- To define a logic we will need four things:
  - ① A language to write formulas
  - ② A notion of truth
  - ③ A notion of what the formulas mean 'model theory'
  - ④ A deduction procedure 'proof theory'
- We will cover each in turn, starting with 'language'
- We have a set  $\Sigma$  called the *alphabet*, it's elements are called *letters*
- Putting letters one after the other we obtain *strings*

# RUDIMENTS OF FORMAL LANGUAGE THEORY

- Given an alphabet  $\Sigma$ , the set of all strings formed from these is denoted  $\Sigma^*$ . There is a special element  $\lambda$  called the *empty string*.
- Length of  $\lambda$  is 0, length of  $a \in \Sigma$  is 1, length of  $\alpha$  denoted  $|\alpha|$  satisfies  $|\alpha\beta| = |\alpha| + |\beta|$
- The main operation on strings is *concatenation* (writing them in sequence). For example, if  $\alpha = abc$  and  $\beta = AB$  then  $\alpha\beta = abcAB$
- Concatenation is *not* commutative,  $\beta\alpha = ABabc \neq \alpha\beta$
- We abbreviate  $\alpha\alpha$  as  $\alpha^2$ , similarly for  $\alpha^3$  etc.
- A **language** over the alphabet  $\Sigma$  is a subset of  $\Sigma^*$
- Since languages are sets, it is meaningful to speak of their union, intersection, and complement (relative to  $\Sigma^*$ )
- The **product** of languages  $R$  and  $S$ , written  $RS$ , is  $\{\alpha\beta \mid \alpha \in R, \beta \in S\}$
- The set  $\cup_{i=0}^{\infty} R^i$  is written  $R^*$  and is called the **Kleene closure** of  $R$ .

# MAIN TYPES OF FORMAL LANGUAGES

- The simplest formal languages are the ones where we list all members explicitly, e.g.  $L = \{a, ab, ba, baab\}$
- **Regular** or **rational** languages are built from these using only Boolean and Kleene operations
- The most complex formal languages we can deal with are called **recursively enumerable**. For each of these, there is an algorithm (Turing machine) that halts whenever the string  $\alpha \in L$  is where we start the computation
- Difference between recursively enumerable and recursive
- We can *prove* the existence of languages that are not r.e. but we cannot *construct* any!
- Summary: there are actually more complex formal languages than we can deal with

# BETWEEN THE EXTREMES

- 1 Between the extremely simple regular (Type 3) and the extremely complex recursively enumerable (Type 0) languages there are intermediary types
- 2 Linear bounded languages (context-sensitive languages) are Type 1
- 3 These require as much memory as the size of the input, but not more. These will be sufficient for logic
- 4 Context-free languages (Type 2) are even less powerful, sufficient for most of logic, except quantifier scope
- 5 Mildly context-sensitive languages (Type 1.5) include indexed grammars, linear indexed grammars
- 6 Complexity of the languages used in logic somewhere between 1 and 1.5