A matematika alapjai, 8. előadás

András Kornai

BMETE91AM35 2022-23 Őszi félév

Óraterv

- Lagrange tétele
- Ezen a héten: nyelvek, automaták, igazság
- Jövő héten: ZH

Rudiments of formal language theory

- Given an alphabet Σ, the set of all strings formed from these is denoted Σ^*. There is a special element λ called the *empty string*.
- Length of λ is 0, length of $a \in \Sigma$ is 1, length of α denoted $|\alpha|$ satisfies $|\alpha\beta| = |\alpha| + |\beta|$
- The main operation on strings is *concatenation* (writing them is 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 α^2, similarly for α^3 etc.
- A **language** over the alphabet Σ is a subset of Σ^*
- Since languages are sets, it is meaningful to speak of their union, intersection, and complement (relative to Σ^*)
- The **product** of languages R and S, written RS, is $\{\alpha\beta | \alpha \in R, \beta \in S\}$
- 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

Rudiments of automata theory

- Finite state automata
- Turing machines

Házi feladatok

- Jövő hétre: CPZ 2.8; 2.26; 2.32; 2.44; 2.59 és a következő HW8.6: Ismerd meg a latex `tabular` környezetet és készíts igazságtáblát a következő oszlopokkal:
  $P, Q, \neg P, \neg Q, P \to Q, \neg Q \to \neg P$
- Csak a pdf kell (a latex forráskód nem) és az email tárgya **Subject: MATALAP NEPTUN HW8**