Foundations of Mathematics, Lecture 9

András Kornai

BMETE91AM35 Fall 2023-24

Contex Free Grammars

- Slightly more powerful than regexps (can do Dyck language)
- In addition to Σ 'alphabet we are interested in' we also have N nonterminal alphabet 'used for scaffolding'
- Start symbol S, rewrite rules $N \to (N \cup \Sigma)^*$
- Rewrite until all scaffolding is removed
- The **yield** of a CFG is the set of strings that can be obtained from a distinguished **start symbol** $S \in V$ by application of productions until no nonterminal is left
- Example: $S \to (S), S \to SS, S \to \lambda$ gives Dyck languge D_2.
- HW9.1 Write the grammar for D_6 using three types of parens ()[]{}

First order language

- It is convenient to use a very large (transfinite) list of *constants*. These are the things we want to talk about (points on the plane, sets, etc.)
- We also permit an infinite, but denumerable list of *variables* $x, y, z, \ldots$ to help us talk about many things at the same time
- Relation symbols, each with a fixed **arity** (the number of factors in the direct product) – again we can permit a more than denumerably infinite supply
- Connectives $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$
- Quantifiers $\forall, \exists$ and brackets [,]

FOL (almost) by CFG

- Ignoring cardinality issues, the nonterminals include WFF, AF, Const, Var, and Rel_n. We will also have some technical symbols , (comma) and $\_$ (placeholder). Brackets, parentheses, connectives and quantifiers are considered terminal symbols, as are the individual constants, variables, and relation symbols
- The rules for **Atomic Formulas**: AF $\rightarrow R_n((\_,)^{n-1}\_)$ 'Each n-ary relation symbol must be followed by (, a string of n empty slots separated by n-1 commas, and terminated by)'
- $\_ \rightarrow c$, $\_ \rightarrow v$ 'Slots of n-ary relational symbols must be filled by constants or variables. So $R(a, x, b)$ is an Atomic Formula, but $S(x, \_)$ is an incomplete atomic formula (doesn't count in the yield, because it still has a nonterminal $\_$)
- To check if a string is an Atomic Formula, you need to check if it starts with a relational symbol, what is the arity of that symbol, and whether the slots are filled by variables/constants

Moving from atomic to more complex formulas

1. WFF $\to$ [AF] 'bracketing an atomic formula gives a well-formed formula'
2. WFF $\to$ [$\neg$ WFF]$||$[WFF $\vee$ WFF]$||$[WFF $\wedge$ WFF]$||$[WFF $\Rightarrow$ WFF]$||$[WFF $\Leftrightarrow$ WFF] 'logical operations on WFFs lead to WFFs'
3. WFF $\to$ [($\forall$ Var) WFF]$||$[($\exists$ Var) WFF] 'quantification'
4. We need to make sure that e.g. $[(\forall x)[(\exists x)R(a, x)]]$ is *not* a WFF 'capturing variables'. This can't be done with a CFG (Type 2) , but very easy with a linear bounded TM (Type 1)
5. HW9.2-4 Write ZFC1,2,5 in FOL Remember $=$ and $\in$ are binary relations
6. HW9.0 Write CFG for the language of arithmetic expressions Use nonterminals Dig (digit), Int (integer), and Nat (natural number)

Truth

- There are two kinds of truth, syntactic and semantic
- We have $\vdash$ 'yields' or 'derives' where $A \vdash B$ means B can be formally derived (proved) from A. For example, in most systems of logic $x = 3 \wedge y = x \vdash y = 3$, but we need a lot of machinery (called *proof theory*) to make this stick. This is pure syntax manipulation: you take formulas and produce new ones by mechanical operations
- We also have $\models$ 'models' where $A \models B$ means that in any model where A is true B is also true. This is more meaningful, but requires *model theory* which spells out the relation between a theory (bunch of formulas) and a set with lots of structure that the formulas are about
- In well-crafted systems $A \vdash B$ implies $A \models B$

The converse is not true!

- In many well-crafted systems (e.g. the first order formulation of Peano Arithmetic) there are statements which are semantically true e.g. PA $\models$ Goodstein's Theorem, but *has no proof there*
- If it has no proof, how do we know it's true? Because in a stronger system (in this case, 2nd order arithmetic) we can prove it
- That the converse is not true for systems endowed with a bit of arithmetic is the celebrated Gödel Incompleteness Theorem
- Our interest here is with the less celebrated, but just as important, Gödel Completeness Theorem
- This says that every formula that is true in all structures is provable
- Wait, how can these both be true? The answer is that PA has more models in first-order axiomatization than in second-order