L20 2022-11-04

Proof Systems

A proof is a sequence of formulas st each line is valid and is an axiom or follows from previous lines by an inference rule.

A proof system specifies which lines are valid, what the axioms are, and what inference rules are.

A PS is decidable if checking wheter a proof is valid is decidable.

A PS is consistent if cannot prove $\varphi \wedge \neg\varphi$.

A PS is complete if for all φ, either φ or $\neg\varphi$ is provable.

lem The set of provable sentences is recognizable.

proof: iterate over proofs in lexicographic order.

This shows we can do step 3.

We can do step 2 in a language that allows string manipulation.

We want to encode $CHECK_A(M,x,y) =$

- $\rightarrow y[0] = (q_0, 0, x)$
- $\rightarrow \forall i \in [1,t]$ $y[i-1]$ valid $\rightarrow y[i]$ valid.
  - i.e.: tape is the same except for pointer transition is correct
- $\rightarrow y[t] = (acc, \cdot, \cdot)$.

But even basic arithmetic theories are incomplete!

Arithmetic Theories

We'll use "Peano arithmetic", which consists of first order logic statements involving $(\mathbb{N}_0, +, \times)$.
(To be precise, we use the weaker "Robinson arithmetic".

Our language has symbols $\neg, \vee, (,), \forall, \exists, +, \cdot, \times, =, 0, s$

Variables range over $\mathbb{N}_0 = 0, 1, 2, \ldots$.

Symbols $\rightarrow, \Leftrightarrow, \leq, 3, x_2, \ldots$ are syntactic sugar.

e.g. $3 = sss0$; $x_2 = xx$.

def arithmetic expression: var / ctnt / expr + expr / expr $\cdot$ expr.

def atom: expr = expr.

def proposition: expr / $\neg$expr / expr $\wedge$ expr / expr $\vee$ expr.

def quantified formula: expr / $\exists x_i$ [qf] / $\forall x_i$ [qf].

def sentence: fully quantified formula.

e.g. $\forall n \exists p [p \geq n \wedge \forall a,b [a \cdot b = p \rightarrow a = 1 \vee b = 1]]$.

How to encode strings with arithmetic?

Gödel's numbering: encode as primes.

e.g. $01101 = 2^0 \cdot 3^1 \cdot 5^1 \cdot 7^0 \cdot 11^1$.