Remark 0.1. We can define \( \text{LenNumOnes}(x) = \text{NumOnes}(|x|) \) = number of 1s in the binary representation of \( |x| \).

But we’ve only defined \( \text{NumOnes}(k) \) for \( k \) such that \( 2^k \) exists (this may not be all \( k \) if we have e.g. a nonstandard model).

Let \( a = (a_l \cdots a_0)_2 \) with \( l = |a| - 1 \). Let

\[
  b_i = \sum_{j=0}^{i} a_i
\]

be the number of 1s in \((a_i \cdots a_0)\). Recall that \( x^* \) is the power of 2 that maximally divides \( x \). Now define

\[
  c_i = \begin{cases} 
    b_i^* & x^* 
    \neq b_{i-1} \text{ or } i = 0 \\
    \epsilon & \text{else.}
  \end{cases}
\]

so that if we code the sequence \((c_l \cdots c_0)\) we use only \( O(|a|) \) qits.

Therefore \( \text{NumOnes}(x) \) is \( \Delta_0 \) defined in \( I\Delta_0 \).

1 Arithmetizing Metamathematics in \( I\Sigma_1 \) (Intensionally)

Remark 1.1. An intensional arithmetizing means that ‘simple’ properties are provable in \( I\Delta_0 \).

If we were to use an extensional method, it could handle Gödel’s Incompleteness theorem for theories as weak as \( \Pi \), but the intensional method only works for things like \( I\Sigma_1 \).

Definition 1.2. We can code formulas into sequences (of integers) and then into integers. This resulting number is the Gödel number of the formula.

The symbols we have are

\[ \land, \lor, \neg, (,), \text{comma}, =, 0, 1, \forall, \exists, \Rightarrow, a, \]

and all the language specific symbols. This lets us write every formula with this finite language \( \Sigma \) (actually more: formulas, sequents, and proofs are all themselves members of \( \Sigma^* \)).

If \( A \) is a string of symbols we write \( \gamma A \gamma \) for the Gödel number of \( A \). We define the property \( \text{WFF}(x) \) if \( x = \gamma A \gamma \) for a well-formed formula \( A \) (can do this with a parse tree etc.).

Remark 1.3. Our goal is to prove theorems about these concepts in \( I\Delta_0 \).

We can define stuff like \( \text{BoundVar}(w) \) easily. It’s a bit trickier to define \( \text{Term}(w) \) where \( w = \gamma t \gamma \) for some term \( t \), but we can do it via recursive parsing. The point is these are all \( \Delta_0 \) definable in \( I\Delta_0 \).

It turns out that \( I\Delta_0 \) can prove ‘simple’ facts such as

\[
  \text{Term}(w_1) \land \text{Term}(w_2) \rightarrow \text{AtomicFormula}(w_1 + \langle \gamma = \rangle + w_2)
\]

and

\[
  \text{WFF}(w_1) \land \text{WFF}(w_2) \rightarrow \text{WFF}(w_1 + \langle \gamma \land \rangle + w_2)
\]

Remark 1.4. Next time we’ll look at quantifiers. These are trickier.