Gödel and Rosser's incompleteness theorems*

In these notes we state and prove Gödel's first incompleteness theorem: every proof system either proves a false mathematical statement or is unable to prove some true statement. Moreover, the statements that satisfy the conditions of the theorem have a simple form: they are fully quantified arithmetical formula, i.e., constructed from Boolean operations, equality, additions, multiplications and quantifiers (see further). Afterwards, we discuss Rosser's variant of the theorem, which is similar, but avoids the notion of mathematical truth. Finally, we show that the set of arithmetical formula without multiplication is decidable.

1 Arithmetical formulas

Arithmetical formulas are formulas that contain variables, have equality relations in them, contain the constants 0, 1, and 2, contain operations for addition + and multiplication (written as concatenation), contain the logical connectives $\wedge$ "and", $\vee$ "or", Not, $\Rightarrow$, and the quantifiers $\forall$ "for all" and $\exists$ "there exists".[1] In short, an arithmetical formula is a *finite* string over the infinite alphabet

$$0, 1, 2, +, =, \neq, (,), \wedge, \vee, \text{Not}, \Rightarrow, \forall, \exists, \overbrace{a, b, c, \dots\dots}^{\text{infinite list of variable names}} \tag{1}$$

such that the operations are correctly formatted. For example, $\exists))+))$ is not correctly formatted, but $\exists a(0 = 0 \Rightarrow 1 + 1 = 1)$ is correctly formatted. A *fully quantified* formula or formula without free variables is a formula in which each variable only appears in the scope of a quantifier. For example the formula $\exists a \exists b\, (ab = 2)$ is fully quantified, but the formula $\exists a(ab = 2)$ not.

We say that a fully quantified formula is *true* if it is true when variables take values over $\mathbb{Z}_{\geq 0} = \{0, 1, 2, \dots\}$, and the operations and logical connectives have their standard meaning. Otherwise, the formula is *false*. For example, $\forall c(F(c))$ is true if the expressions $F(0), F(1), \dots$ are all true.

Examples. The formulas $0 = 1$, $\exists a(aa = 2)$, and $\forall a \exists b(b + 1 = a)$ are false formulas, because no square root of 2 is integer and no $b \in \mathbb{Z}_{\geq 0}$ exists such that $0 = b + 1$. Now we consider the

*Theoretical Computer Science 2017-2018, National Research University Higher School of Economics. Thanks to Kirill Neklyudov for reporting mistakes in earlier versions.
First version: 21 Febr 2018.
Update May 22, 2018: added and improved explanations in many places.

[1] Formally, arithmetical formulas are defined as the first order language that contains one binary predicate symbol (equality), two constants (0 and 1), and two binary function symbols (addition and multiplication).

following formulas:

$$\forall a\,(0 = 1 \Rightarrow \exists b(bb = 2))$$
$$\forall a \forall b\,(aa - bb = (a + b)(a - b))$$
$$\forall r \forall s\,(1 + 1 + 1 = rs \Rightarrow (r = 1 \vee s = 1))$$
$$\forall m \exists k \forall r \forall s\,(m + k = rs \Rightarrow (r = 1 \vee s = 1))\,.$$

The first is true because the condition of the implication is false. The second is a well known algebraic identity. The third states that if the product rs equals 3, then either $r = 1$ or $s = 1$. In other words, 3 is a prime number. The last one states that there exist infinitely many prime numbers.

Remark. Several symbols in this language are redundant, for example, in a formula we can equivalently replace any symbol 2 by $(1 + 1)$. This symbol is conveniently added to write formulas shorter. The same is true for $\Rightarrow$, because any subformula of the form $F \Rightarrow G$ can be replaced by $(\text{Not}\, F) \vee G$. Also $\neq$ is redundant, because we can replace $s \neq t$ by $\text{Not}(s = t)$.

Despite the simplicity of this language, we can directly formulate profound questions of mathematics. For example, Goldbach's conjecture states that every even integer larger than 2 is the sum of two primes. This corresponds to the formula

$$\forall k \exists p \exists q \Big(2(k + 2) = p + q \wedge \forall r \forall s \big((rs = p \vee rs = q) \Rightarrow (r = 1 \vee s = 1)\big)\Big).$$

Whether this formula is true is perhaps the oldest and most well-known open problem in mathematics.

Exercise 1. Lagrange's four square theorem states that every nonnegative integer is the sum of 4 squares of integers. Express this statement as an arithmetical formula.

2 The set of true arithmetical formulas is undecidable

We study the following problem.

Input: An arithmetical formula.
Question: Is the formula true?

Gödel's incompleteness theorem is a corollary of the undecidability of the problem above. To prove this, we will reduce the halting problem $\text{H}_{\text{TM},\varepsilon}$ to the set of true arithmetical formulas. In other words, for each Turing machine we construct a formula that is true if and only if the machine halts on empty input. This is remarkable, because it means that we can somehow "simulate" any arbitrarily long computation using a formula with finitely many quantifiers, additions and multiplications. Before presenting the construction, we need to get more familiar with arithmetical formulas, and in the following two lemmas we show that many other mathematical operations are expressible with them.

In the construction of bigger formula we consider expressions with free variables. The 3 formulas

$$a = b \Rightarrow aa = bb$$
$$\exists c(a + c = b)$$
$$\exists c(a + 1 + c = b)$$

each have free variables a and b. The first formula is true for all values of a and b, the second is true for $a \leq b$ and the third is true for $a < b$.

Exercise 2. Let $\lfloor r \rfloor$ denote the largest integer that is at most r. Let $b \bmod c$ represent the remainder of the division of b by c. Thus, $0 \leq b \bmod c < c$. Write arithmetical formulas with free parameters a, b and c that are equivalent to

- $a = b \bmod c$,
- $\lfloor a/b \rfloor = c$.

The exercise above allows us to transform formulas with modulo operations to arithmetical formulas. We explain this with an example. Let $F(x)$ be an arithmetical formula with a free variable x. The formula $F(b \bmod c)$ is equivalent to $\exists a\big(F(a) \wedge (a = b \bmod c)\big)$. In this expression we can expand the right hand side of the conjunction using the solution of the exercise. More generally, this can be done for all occurrences of modulo operations.

Let us also look at some formulas that are not arithmetical. The expression $\exists \ell (a = 2^\ell)$ is true if a is a power of 2, but this is not an arithmetic formula, because it uses the power operation. It is not obvious how we can rewrite this as an arithmetic formula. The following formula is also *not* arithmetical:

$$\exists n \exists \ell_1 \exists \ell_2 \ldots \exists \ell_n \big(\ell_1 = 2 \ \wedge \ 2\ell_1 = \ell_2 \ \wedge \ 2\ell_2 = \ell_3 \ \wedge \ \ldots \ \wedge \ 2\ell_{n-1} = \ell_n \ \wedge \ \ell_n = a\big).$$

Why? We do not have a symbol "..." in our alphabet, neither do we have indexed variables. Every arithmetical formula has a fixed number of quantifiers, and the list $\ell_1, \ldots, \ell_n$ can have any length. However, there is still a method to express powers of 2 using arithmetical formula. This will follows by substituting $b = 2$ in the following lemma.

Lemma 1. *The formula*

$$b \neq 0 \ \wedge \ b \neq 1 \ \wedge \ \forall d \forall s \Big(ds = B \Rightarrow (d = 1 \vee \exists e (eb = d)) \Big)$$

is true if and only if $b \geq 2$ and the set of divisors of B equals $\{1, b, b^2, \ldots, b^p\}$ for some p.

Proof. The upward implication is easy. We prove the downward implication in detail.

First we show that if 0 is a divisor of B, then the formula is false. Note that 0 does not divide any $B \geq 1$. Thus this implies $B = 0$. Hence, every d is a divisor of B, and in particular, $d = b + 1$ is a divisor. But b does not divide $b + 1$ if $b \geq 2$. The implication and the formula are false.

Assume the formula is true. Clearly $b \geq 2$. We need to show that every divisor of B equals b^m for some m. Let d_1 be a divisor of B. Our assumption implies $d_1 \neq 0$. Let s be such that $d_1 s = B$. The formula implies that either $d_1 = 1$ or that $d_1 = eb$ for some e. In the first case, d_1 is a power of b (recall that $b \neq 0$). In the second case we repeat the argument for $d_2 = e = d_1/b$. Note that d_2 is also a divisor of B, and $d_2 < d_1$ because $b \geq 2$. We can now repeat the argument for d_2 and conclude that either $d_2 = 1$ or find a strictly smaller divisor d_3. After finitely many repetitions we find some $d_m = 1$ and we conclude that $d_1 = b^{m-1}$. $\square$

Definition 2. *A relation* $R \subseteq \mathbb{Z}_{\geq 0} \times \mathbb{Z}_{\geq 0}$ *is* arithmetical *if there exists an arithmetical formula* F *with* 2 *free variables* a *and* b *such that* $F(a, b)$ *is true if and only if* $(a, b) \in R$. *In a similar way, arithmetic relations over* $\mathbb{Z}_{\geq 0}^k$ *are defined by formulas with* $2k$ *free variables.*

A relation $E \subseteq V \times V$ defines a directed graph with self-loops. A path is defined in the usual way, and we also consider paths of length 0, that connect every vertex to itself.

Definition 3. *The* transitive closure *of a relation $E \subseteq V \times V$ is given by the relation*

$$\{(v, w)\colon \text{ there exists a path from } v \text{ to } w \text{ in the graph } (V, E)\}.$$

For example, the relation $\{(m, n)\colon n = m + 2\}$ over $\mathbb{Z}_{\geq 0}$ has transitive closure

$$\{(m, n)\colon \exists k \in \mathbb{Z}_{\geq 0}(n = m + 2k)\}.$$

Notice that both relations are defined by simple arithmetical formula.

Lemma 4. *If a relation is arithmetical then its transitive closure is also arithmetical.*

Exercise 3. Use Lemma 4 to show that there exists an arithmetical formula $F(b, c)$ that is true if and only if c is a power of b.

Proof of Lemma 4. We first prove the lemma for relations over $\mathbb{Z}_{\geq 0}$, afterwards we modify the proof for relations over $\mathbb{Z}_{\geq 0}^k$ for any k. Let R be an arithmetical formula with 2 free variables. For each $u, v \in \mathbb{Z}_{\geq 0}$ we need a formula with the meaning:

$$\begin{aligned} &\text{“There exists a list } \ell_1, \ldots, \ell_n \text{ with } \ell_1 = u \text{ and } \ell_n = v \\ &\quad \text{such that } R(\ell_i, \ell_{i+1}) \text{ holds for all } i = 1, \ldots, n-1.\text{”} \end{aligned} \tag{2}$$

Our approach is inspired by how we write numbers in some base b. If the expansion of a number L in base b is $\ell_n \ell_{n-1} \ldots \ell_2 \ell_1$, then we have $L = \sum_{i=1}^{n} \ell_i b^{i-1}$. Thus the ith digit of L's expansion equals $\ell_i = \lfloor L/b^{i-1} \rfloor \bmod b$, where $\lfloor r \rfloor$ is the largest integer that does not exceed $r \in \mathbb{R}$. From now on we only use integer devision, and omit $\lfloor \cdot \rfloor$. Let us rewrite the statement above.

$$\begin{aligned} \exists n \exists b \exists L \Big[& L \bmod b = u \ \wedge \ L/b^n \bmod b = v \\ & \wedge \ \forall m \Big(m < n \ \Rightarrow \ R\left(L/b^m \bmod b,\, L/b^{m+1} \bmod b\right) \Big) \Big] \end{aligned} \tag{3}$$

To prove that (3) implies (2), simply choose $\ell_i = L/b^{i-1} \bmod b$. To prove that (2) implies (3), choose some b larger than any of the elements $\ell_1, \ldots, \ell_n$ and choose $L = \sum_{i=1}^{n} \ell_i b^{i-1}$. Note for later use, that this implication also holds if in (3) we additionally require that b is a prime. Thus, (3) is equivalent after adding: $\wedge$ “b is prime” inside the square brackets.

From the discussion below exercise 2, we know that the occurrences of expressions of the form $\lfloor L/c \rfloor = a \bmod b$ can be transformed to arithmetical operations. Thus the statement above is almost arithmetical. The only problem is that we do not have expressions for powers. But note that the only values that can appear as a power are $\{1, b, b^2, \ldots, b^n\}$. If b is prime, we can write this set as divisors of a fixed power of b.

$$\begin{aligned} \exists b \exists B \exists L \Big[& L \bmod b = u \ \wedge \ L/(Bb) = v \\ & \wedge \ \text{“}b \geq 2 \text{ and the divisors of } B \text{ are } \{1, b, \ldots, b^p\} \text{ for some } p\text{”} \\ & \wedge \ \forall d, d' \left(dd' = B \ \Rightarrow \ R\Big(L/d \bmod b, L/bd \bmod b\Big)\right) \Big]. \end{aligned} \tag{4}$$

To prove that (3) implies (4), we start with the equivalent variant of (3) in which b is a prime. We choose $B = b^{n-1}$. The only divisors are of the required form. To prove that (4) implies (3), we choose $n = p + 1$ for p such that $B = b^p$.

Finally, we use Lemma 1 and exercise 2 to rewrite this expression in arithmetical form. This finishes the proof for relations over $\mathbb{Z}_{\geq 0}$.

For relations R over $(\mathbb{Z}_{\geq 0})^k$ the formula is similar: elements of the arithmetic closure of R are characterized by a list of k-tuples so that R is true for each pair of subsequent tuples. In this case, we follow the proof above, but now we use k numbers $L_1, \ldots, L_k$ whose representations in base b encode the k-tuples. □

We are now ready to prove the main result of this section.

Theorem 5. *The set of true arithmetical formulas is undecidable.*

Proof. As discussed above, we reduce the halting problem $\mathrm{H}_{\mathrm{TM},\varepsilon}$ to the set of true arithmetical formula. Thus, we start with a description of a Turing machine, and compute a formula that is true if and only if the machine halts on empty input. The construction happens in several steps. First we encode machine states using 3 integers. Then we define an arithmetical relation on triples of integers that encode subsequent states in a computation. We apply Lemma 4 to obtain a formula that is true if one state can be reached by the other after any number of computation steps. From this we easily get the required formula.

Let $(Q, \Gamma, d, q_{\text{start}})$ be a Turing machine, as we defined earlier in the lecture notes. We assume that $|\Gamma| \geq 2$ and that the machine can only halt by falling of the tape. If our machine does not satisfies these conditions, than we first transform it to such a machine with the same input output relation. This assumption is convenient at the end of the proof.

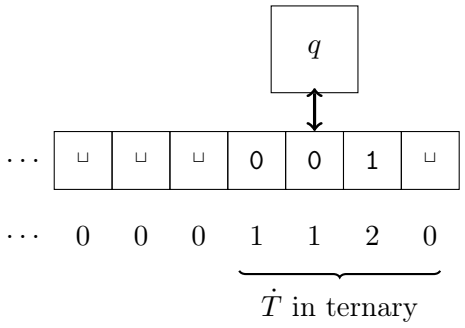

Figure 1: Coding a tape as an integer in base $|\Gamma|$.

We map the tape symbols Γ to the numbers $\{0, 1, \ldots, |\Gamma| - 1\}$ such that $\sqcup$ is mapped to 0. We draw the infinite end of the tape to the left. (This has no mathematical consequences.) We can now interpret a tape as a number in base $|\Gamma|$ by ignoring all leading zeros. See figure 1 for an example where $\Gamma = \{\sqcup, \mathtt{0}, \mathtt{1}\}$ and where the symbols $\sqcup, \mathtt{0}$, and $\mathtt{1}$ are represented by $0, 1$ and 2.

Initially, $T = \sqcup^\infty$, and this number equals 0. At each moment in a computation, the tape contains finitely many non blank symbols, and hence this number is finite. For a tape $T \in \Gamma^\infty$ we write this number as $\dot{T}$.

Recall from our definition of a Turing machine that a machine state is given by a triple $(i, q, T) \in \mathbb{Z}_{\geq 0} \times Q \times \Gamma^\infty$, and that a position $i = 0$ represents a computation head that has fallen of the tape. We encode this state as the triple of integers given by

$$\left(\left\lfloor |\Gamma|^{i-1} \right\rfloor, \dot{q}, \dot{T} \right),$$

where $\dot{r} \in \mathbb{Z}_{\geq 0}$ represents a unique integer for each control state $r \in Q$. For example, the start state is encoded by $(1, \dot{q}_{\text{start}}, 0)$. If $i = 0$, the first integer in this triple equals 0, because of the assumption $|\Gamma| \geq 2$ and downward rounding.

How are the numbers updated when the machine performs a computation step? The formula for $T = T_1 T_2 \ldots$ is:

$$\dot{T} = \sum_{i=1}^{\infty} \dot{T}_i |\Gamma|^{i-1}.$$

A computation step in which the computation head has state q, reads a, writes c, moves left, and goes to control state r, corresponds to a transformation

$$\left(I, \dot{q}, \dot{T} \right) \quad \mapsto \quad \left(I|\Gamma|, \dot{r}, \dot{T} + (c - a)I \right)$$

for some values of I and $\dot{T}$. Now we can write the relation $F_{\text{left}}(I, \dot{q}, \dot{T}; J, \dot{r}, \dot{S})$ that checks whether a state $(I, \dot{q}, \dot{T})$ can be transformed to $(J, \dot{r}, \dot{S})$ in 1 computation step by a left move:

$$\bigvee_{\substack{u,a,c,v: \\ d(u,a)=(c,L,v)}} \left(\dot{u} = \dot{q} \quad \wedge \quad \dot{a} = \dot{T}/I \bmod b \quad \wedge \quad \dot{T} + cI = \dot{S} + aI \quad \wedge \quad J = |\Gamma| I \right)$$

This is an arithmetical formula, because the disjunction has at most $|Q| \cdot |\Gamma|$ terms. Recall that for each machine we have to build a separate formula, thus the machine is fixed.

For a right move we make a similar formula, but now we use the condition $J = |I|/|\Gamma|$. By exercise 2 this is expressible as an arithmetical formula. We use integer division rather than $|\Gamma| J = I$, because in a computation step in which the machine falls of the tape, the value of J corresponds to $1/|\Gamma| = 0$, (recall that $|\Gamma| \geq 2$).

The formula $F(I, \dot{q}, \dot{T}; J, \dot{r}, \dot{S}) = F_{\text{left}}(\ldots) \vee F_{\text{right}}(\ldots)$ is true if and only if the machine can go in 1 computation step from $(I, \dot{q}, \dot{T})$ to $(J, \dot{r}, \dot{S})$.

We apply Lemma 4 to the relation given by F, and obtain a relation G that is true if a machine can go from one state to another after finitely many computation steps. The formula

$$\exists \dot{r} \exists \dot{S} \Big(G(1, \dot{q}_{\text{start}}, 0; 0, \dot{r}, \dot{S}) \Big)$$

is true if and only if the machine halts.

Note that all steps in the construction here and in the proof of Lemma 4 are computable. Hence, $\text{H}_{\text{TM},\varepsilon}$ reduces to the set of true arithmetical formula. Thus the latter set is undecidable. □

Remark: In the proof of Rosser's theorem below, we consider the set of all triples $\langle M, n, m \rangle$ such that if machine M is started with a tape $\dot{T} = n$, it halts with a tape $\dot{S} = m$. The corresponding formula is $\exists \dot{r}(G(1, \dot{q}_{\text{start}}, n; 0, \dot{r}, m))$

Exercise 4. Encode machine states as in the proof of Theorem 5. Write down an arithmetical formula F that is true if and only if the machine is above a cell at an even position that contains a blank.

Exercise 5. Imagine that in the definition of an arithmetical formula we restricted the number of different variables names to a large constant, say 100. In other words, we define arithmetical statements on the alphabet given by equation (1) where the number of variable names is precisely 100. Is Theorem 5 still true?

3 Gödel's first incompleteness theorem

This theorem states that if a proof system only proves true formulas, then there must exist some true arithmetical formula that it can not prove. We will not formally define "proof systems", because it is impossible to give any realistic example in a few paragraphs. Fortunately, this is not needed. We only need one property: the set of provable formulas in a proof system is recognizable. We consider proof systems simply as a device that receives a mathematical statement and a proof, and checks whether the given proof is a correct proof of the statement. If so, it outputs `true`, otherwise `false`.

Definition 6. *A* proof system *is a computable binary predicate P mapping an arithmetical formula F and a* proof x *(which can be any string over some* proof alphabet*) to* $\{\texttt{true}, \texttt{false}\}$. *We say that P* proves *F if there exists a string x such that $P(F, x)$ is true. Such formulas F are called* provable.

With this definition, the set of formulas that have a proof, is recognizable. Indeed, one obtains a machine that halts on input F if and only if F is provable by simply search for a valid proof, more precisely: evaluate $P(F, x)$ for all proof strings x in lexicographic order (or any other order) and halt if an x is found for which $P(F, x)$ is true.

Corollary 7 (Gödel 1931)**.** *Every proof system proves some false arithmetical formula or is unable to prove some true arithmetical formula.*

Proof. The negation of this claim is that there exists a proof system P that proves precisely all true formulas. Let us assume this is true. Let F be an arithmetical formula. By definition of mathematical truth, either F or its negation $\text{Not}(F)$ is true. By assumption, the true one is provable and the other not. Thus, to find the true one, we just search for a proof of both F or $\text{Not}(F)$: for all x (say in lexicographical order) we evaluate $P(F, x)$ and $P(\text{Not}(F), x)$ until an x is found that makes one of the predicates true. This might take a long time, but such a correct proof x always exists and the search always terminates. Hence, the set of true arithmetical formulas is decidable. This contradicts Theorem 5: our assumption must be false and the corollary must be true. □

It is interesting to prove this corollary in a more direct (but longer) way. We use a variant of the liar paradox: "This is a lie." The idea is to consider the following statement:

The statement in the box has no proof.

Suppose the statement has a proof. Then the proof system proves a false statement. Otherwise, it is true, and hence unprovable by assumption. In both cases it satisfies the conditions of Corollary 7.

How can we make an arithmetical formula that has the same meaning as the statement above? Let us start with a simpler question: given a formula F, how can we write down a formula G_F that is true if and only if F has no proof? For this, we use a Turing machine U that on empty input searches for a proof of F. Upon finding such a proof, it halts and if no proof exists, it runs forever. From the proof of Theorem 5 we obtain an arithmetical statement G_F that is true if and only if U does not halt, i.e., if and only if F has no proof.

Now, we must find a way to let a formula refer to itself. In other words, we must find a formula F such that $G_F = F$. For this[2] we adapt the construction of G by considering formulas with a free variable.

Consider a list of all arithmetical formulas with a free variable n, in some order (say, the lexicographic order):

$$F_1(n),\ F_2(n),\ F_3(n),\ \ldots$$

Let V be a Turing machine that on input an integer N, computes F_N, replaces the free variable with the value N, searches for a proof, and halts if such a proof is found. Thus $V(N)$ halts if and only if $F_N(N)$ is provable.

Let $G(n)$ be a formula with free parameter n that is true precisely for those values n for which $V(n)$ does not halt. Such a formula exists by applying the remark after Theorem 5 and negating the result. Thus, for all $N \in \mathbb{Z}_{\geq 0}$:

$$\text{“}G(N) \text{ is true”} \quad \Longleftrightarrow \quad \text{“}F_N(N) \text{ is } \mathbf{un}\text{provable”}$$

Because $G(n)$ has the free parameter n, it appears in the list $F_1(n), F_2(n), \ldots$ Let N be its position, thus $F_N(\cdot) = G(\cdot)$ and therefore

$$\text{“}F_N(N) \text{ is unprovable”} \quad \Longleftrightarrow \quad \text{“}G(N) \text{ is unprovable”}$$

The two equivalences combined state that the formula $G(N)$ is true if and only if it is unprovable. This implies Gödels theorem.

4 Rosser's theorem

Gödel's theorem argues that no proof system proves precisely those formula that are true. But perhaps, being always correct might be a too strong assumption. The example with self-referential statements seems very strange, how can the mathematical truth of such statements be relevant in practice? Perhaps, we might be happy with a proof system that proves some false theorems, as long as it does not contradict anything observable or anything else that we have proven. [3]

[2] The fixed-point theorem implies that such formula exists for every mapping G. This is the approach of Sipser's book. We will prove this in a direct way.

[3] The proof of Matiyasevich theorem implies that for each proof system there exists a polynomial P over several variables such that the self-referential statement of the previous paragraph is equivalent to the statement “P has no integer solutions”. Thus if we allow the proof system to prove false theorems, we can not even trust its claims about polynomials.

However, we should be careful. Many proof systems allow to prove theorems by contradiction. If there is only 1 provable formula F for which also $\mathrm{Not}(F)$ is provable, it would make all formulas immediately provable, and hence the proof system would be completely useless. Thus, if we weaken the requirement of correctness in Gödel's theorem, it is natural to require the system to be contradiction free.

Rosser's theorem claims that even for contradiction-free proof systems, there are always sentences that can neither be proven nor disproven.

Definition 8. *A* disproof *of an arithmetical formula F is a proof of its negation* $\mathrm{Not}(F)$*. A proof system is* consistent *if no formula has both a proof and a disproof.*

We discuss Rosser's statement in a bit informal way.

For every consistent and sufficiently powerful proof system
there exists an arithmetical formula that has neither a proof nor a disproof.

It is hard to define "sufficiently powerful" precisely. In Rosser's original theorem it is required that the proof system should extend some very basic proof system.[4]

We already discussed that we can write statements of the form "on input n machine M outputs m" as an arithmetical formula $F_{M,n,m}$. See the remark after the proof of Theorem 5. Let us say that a proof system is *sufficiently powerful* if it can prove all such true statements, roughly stated, if proofs can represent transcripts of halting computations on a Turing machine.

Proof. Consider a Turing machine M that computes a partial function from $\mathbb{Z}_{\geq 0}$ to $\{0,1\}$ that has no (total) computable extension. Such machine exists by exercise 5 in the notes *undecidable sets*. Let

$$f(n) = \begin{cases} 1 & \text{if } F_{M,n,1} \text{ is provable,} \\ 0 & \text{if } \mathrm{Not}(F_{M,n,1}) \text{ is provable.} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

If the proof system is consistent, then f is well defined. If it is sufficiently powerful, then $M(n) = 1$ implies $f(n) = 1$ and $M(n) = 0$ implies $f(n) = 0$. Thus f is an extension of the partial function defined by M. If every formula is provable or disprovable, then f is total and computable.

In summary, f is a computable extension of M's partial function and this contradicts the choice of M. Our version of Rosser's theorem is proven. □

It is possible to give a proof as for Gödel theorem using some variant of the liars paradox? We present the high level idea. Consider the following:

For every proof of this statement there exists a shorter disproof of this statement.

[4] A more precise statement can be found here Theorem 4.6.2 page 111 in
`http://www.andrew.cmu.edu/user/avigad/Teaching/candi_notes.pdf`

We argue that if the statement above has a proof or a disproof, then the proof system is inconsistent.

Assume the statement has a proof. Then, we can check whether there exists a disproof among the finitely many proof-strings that are shorter. If we find such a disproof, we conclude that the proof system is inconsistent. If no such disproof exists, the finite check gives us a disproof of the statement in the box above (here we use that the proof system is "sufficiently powerful" so that it can verify such a check). Again, this implies that the proof system is inconsistent.

Now assume the statement above has a disproof. The negation is: "There exists a proof of the statement in the box without a shorter disproof." In particular, there exists at least one proof of the statement. Again the proof system is inconsistent. Rosser's claim is proven.

5 A decidable theory

Theorem 9. *The set of true arithmetical formula without multiplications is decidable.*

This theorem is proved using automata. In the following exercises we show that the set of true arithmetical formulas without multiplications is decidable. In stead of solving the exercises below, you could also read the proof of Theorem 6.12 on p255 in Sipser's book.

Exercise 6. This is exercise 1.37 p89. Let

$$\Sigma_3 = \left\{ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \ldots, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\}$$

be the set of all column matrices of size 3 with elements in $\{0, 1\}$. A string of Σ_3 elements gives three rows of elements in $\{0, 1\}$. Interprete these rows as numbers in binary, and let

$$B = \{w \in \Sigma_3^* : \text{ the bottom row is the sum of the two top rows}\}$$

For example:

$$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \in B \qquad \text{but} \qquad \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \notin B$$

Show that this set is regular. Hint: first show that the set of all reversed strings of B is regular.

Definition 10. *For all $k \geq 1$, let Σ_k be the set of all column matrices of height k. We say that a set $S \subseteq \mathbb{Z}_{\geq 0}^k$ is* regular *if there exists a nondeterministic automaton that accepts precisely the words $w \in (\Sigma_k)^*$ whose k rows represent a k-tuple in S in binary (as in the previous exercise). A formula ϕ with free parameters $a, b, \cdots, e$ is regular if the set of values for which it is true, is regular.*

Exercise 7. Assume that ϕ and φ are regular. Show that also $\phi \wedge \varphi$, $\phi \vee \varphi$ and $\text{NOT}(\phi)$ are regular. Assume that ϕ is regular. Let ψ be the formula given by $\psi(a, b, \ldots, e) = \exists x \phi(x, a, b, \ldots, e)$. Show that also this ψ is regular. Can you prove the same for the $\forall$-quantifier?

Exercise 8. Show that the set of true arithmetical formula without multiplications is decidable. For this, reduce this set to the set of nondeterministic automata that accept the empty string. Consider a more general mapping from formula with k free parameters to automata over the alphabet Σ_k. Use induction on the number of $\forall, \exists, \wedge, \vee, \mathrm{NOT}$ operations.

Solutions

Exercise 1. $\forall n \exists a \exists b \exists c \exists d \big(n = aa + bb + cc + dd\big)$

Exercise 2.

- $\big(\exists d(a + d + 1 = c)\big) \ \wedge \ \big(\forall k \forall r(a = kc + r \Rightarrow r = b)\big)$
- $\forall k \forall r \forall s \big(a = ck + r \Rightarrow (r + s \neq b)\big)$

Exercise 3. Consider the arithmetical relation $E \subseteq \mathbb{Z}_{\geq 0} \times \mathbb{Z}_{\geq 0}$ given by

$$(a, \tilde{a}) \in E \quad \Longleftrightarrow \quad \tilde{a} = ba.$$

Its arithmetical closure contains precisely the pairs (a, c) for which $c = b^k a$ for some k. By Lemma 4, there exists an arithmetical formula $G(a, c, b)$ that is true precisely for these pairs (a, c). Thus, c is a power of b if and only if $G(1, c, b)$ is true.