# 2. Abstract views of incompleteness

## 2.1. Models of computation: recursive functions

Computability and recursion are often identified. However in principle they are different concepts (see Soare (1996)): computability is a more general term, whereas the primary mathematical meaning of *recursion* has always been *definition by induction* (i.e., by recursion), namely defining a function $f$ at an argument $x$ using its own previously defined values. In the seminal paper Gödel (1931) the great Czech logician used the notion of a *primitive recursive function* (called *rekursive Funktion*) because these functions were easily representable in formal system PA for arithmetic, and were sufficient to enable him to code all the syntactic objects so that he could obtain self-reference and thereby incompleteness. The main property of this class of functions is the primitive recursion scheme, which yields an *inductive* definition of $f(n+1)$ using the preceding value $f(n)$ and previously defined functions $g$ and $h$. Gödel realised, however, that the primitive recursive functions did not include all effectively computable functions, and in 1934 he proposed a wider class of functions based on an earlier suggestion of Herbrand. Gödel called these the *general recursive functions.*

Actually in the initial study of computability from 1931 to 1937 researchers considered only *total* computable functions. It was Kleene in 1938 who first proposed considering *partial* computable functions. Hence Kleene extended the class of primitive recursive functions adding a scheme of *Unbounded Search*, to introudce $\mu - recursive$ *partial* functions, i.e. the computation model whose fundamental concepts we will illustrate in this chapter (see again Soare (1996)).

**Definition 3.** *A function $f$ from natural numbers to natural numbers is said partial if $Dom(f) \subseteq \mathbb{N}$; if in particular $Dom(f) = \mathbb{N}$, then the function is total.*

**Definition 4.** *A set $X \subseteq \mathbb{N}$ is called recursive iff its characteristic function is recursive:*

$$\chi_X(x) = \begin{cases} 1 & if\, x \in X \\ 0 & if\, x \notin X \end{cases}$$

*is total recursive (that is computable). Moreover, $X$ is called recursively enumerable (or computably enumerable) if either $X = \emptyset$, or coincides with the set of values of a total recursive function.*

Examples of recursive sets are $\mathbb{N}$, $2\mathbb{N}$, finite sets, cofinite sets; moreover. if $X, Y$ are recursive, then also $\overline{X}$, $X \cap Y$, $X \cup Y$, $X \smallsetminus Y$ are recursive. Note the following difference:

1. The set of numbers $n$ for which there exists a sequence of *exactly* $n$ occurrences of the number "7" in the expansion of $\pi$ is recursively enumerable: we generate the expansion $\pi$; every time we see a string of a certain number of digits "7", we count it and add the number to the list.
2. The set of numbers $n$ for which there exists a sequence of *at least* $n$ occurrences of the number "7" in the expansion of $\pi$ is recursive: if $n$ is in this set, then we know that at least $n$ occurrences of "7" in $\pi$ appears and therefore every $k < n$ is in turn in this set. It follows that such a set, either is all $\mathbb{N}$, or an initial segment of it. In both cases it is recursive.

Duccio Pianigiani, University of Siena, Italy, duccio.pianigiani@unisi.it, 0000-0001-9441-7226

Referee List (DOI 10.36253/fup_referee_list)
FUP Best Practice in Scholarly Publishing (DOI 10.36253/fup_best_practice)

Duccio Pianigiani, *Abstract views of incompleteness*, © Author(s), CC BY-SA 4.0, DOI 10.36253/979-12-215-0778-2.05, in Duccio Pianigiani, *Lectures in Proof Theory and Complexity*, pp. 33-60, 2025, published by Firenze University Press and USiena PRESS, ISBN 979-12-215-0778-2, DOI 10.36253/979-12-215-0778-2

Book References DOI 10.36253/979-12-215-0778-2.references

The class of primitive recursive functions has several fathers and mothers, from Dedekind to Skolem to Gödel. However it was Péter (1932) who defined primitive recursive the class of functions introduced by Gödel and studied this class of functions in depth.

**Definition 5.** *The class of primitive recursive functions is generated by the following axioms:*

1. *(Zero)* $Z(x) = 0$
2. *(Successor)* $S(x) = x + 1$
3. *(Projections)* $P_i^{n+1}(x_0, ..., x_n) = x_i$
4. *(Composition)* $f(x_0, ..., x_m) = g(h_0(x_0, ..., x_m), ..., h_n(x_0, ..., x_m))$, *where* $g, h_0, ..., h_n$ *are primitive recursive.*
5. *(Primitive recursion)*

   *(a)* $f(x_0, ..., x_m, 0) = g(x_0, ..., x_m)$

   *(b)* $f(x_0, ..., x_m, n + 1) = h(x_0, ..., x_m, n, f(x_0, ..., x_m, n))$

   *where* $g, h$ *are primitive recursive.*

Many of the most familiar functions are primitive recursive. For example the famous Fibonacci function:

1. $f(0) = 0$
2. $f(1) = 1$
3. $f(n + 2) = f(n + 1) + f(n)$

Although at first glance it does not seem (to step 3. we have used two previous values of $f$, and not one), it is primitive recursive: the recursion scheme "on the course of values" used there, does not get out of primitive recursive functions.

Often are thus considered most complicated scheme of recursion, that however does not get out of primitive recursive functions. Strange as it may seem, the following scheme of double recursion is primitive recursive:

1. $\phi(0, n) = g(n)$
2. $\phi(m + 1, 0) = h(m)$
3. $\phi(m + 1, n + 1) = \psi(m, n, \phi(m, \gamma(m, n)), \phi(m + 1, n))$

where $g(x)$, $h(x)$, $\psi(x, y, u, v)$ and $\gamma(u, v)$ are given primitive recursive functions. However, note that in the right side of the definition, at point 3 in the first occurrence of $\phi(x, y)$ has been substituted in place of $y$ a function $\gamma(u, v)$ given at the beginning. Compare this case with the next case (Ackermann-Péter function). The scheme ("nested recursion") applied in the definition of this function is not reducible to the primitive recursion (note that in the third clause of the definition, the $A$ occurs twice, in two nested occurrences):

1. $A(m, 0) = m + 1$
2. $A(0, n + 1) = A(1, n)$
3. $A(m + 1, n + 1) = A(A(m, n + 1), n)$

For instance, $A(0, 1) = 1$, $A(3, 3) = 61$, $A(4, 4) = 2^{2^{2^{65536}}}$ ... The function $A(x, y)$ majorizes each primitive recursive function, i.e., for each $f$ primitive recursive function, there exists a number $n$ such that: $f(x_0, ..., x_k) < A(max\{x_0, ..., x_k\}, n)$. It follows that the Ackermann function is not primitive recursive: in fact, if it were, it would be primitive recursive also $f(n) = A(n, n) + 1$; therefore, there would exist a $k$ such that $f(m) < A(m, k)$, hence $A(k, k) + 1 = f(k) < A(k, k)$, a contradiction. After the discovery of this function, a problem arose for Hilbert's school: is Ackermann's function finitist?

> We may however admit certain extensions of the scheme of recursion as well as the induction schema, without taking away what is characteristic of the method of recursive number theory (Hilbert and Bernays (1934) vol. I, p. 325).

> The nested recursion...appear to me to be finite in the same sense as primitive recursion, i.e. of one regards them as a statement of a computation procedure where one can recognize that the function defined by the respective process satisfies the recursion equations (Bernays (1970))

Subsequently, important logicians as Simpson and Tait have proposed to identify the finitary mathematics with Skolem's *Primitive Recursive Arithmetic* PRA. Kreisel proposed instead the identification of finitist functions with those provably total in PA. The detailed historical analysis conducted by Zach (1998) showed how Hilbert and Bernays regarded finitist arithmetic as partially but not necessarily completely formalised by primitive recursive arithmetic.

Here some example of primitive recursive functions.

1. *Sum.*
   (a) $x + 0 = x$
   (b) $x + (S(y)) = S(x + y)$
2. *Predecessor.*
   (a) $P(0) = 0$
   (b) $P(y + 1) = y$
3. *Truncated difference.*
   (a) $x \dot{-} 0 = x$
   (b) $x \dot{-} (S(y)) = P(x \dot{-} y)$
4. *Multiplication.*
   (a) $x \cdot 0 = 0$
   (b) $x \cdot (S(y)) = (x \cdot y) + x$
5. *Exponentiation.*
   (a) $x^0 = 1$
   (b) $x^{y+1} = x^y \cdot x$
6. *Absolute value.* $|x - y| = (x \dot{-} y) + (y \dot{-} x)$
7. *Factorial.*
   (a) $0! = 1$
   (b) $(y + 1)! = y! \cdot (y + 1)$
8. *Minimum* $min\{x, y\} = x \dot{-} (x \dot{-} y)$ and *Maximum* $max\{x, y\} = x + (y \dot{-} x)$.
9. *Sign functions*:
   (a) $sg(0) = 0$
   (b) $sg(x + 1) = 1$
   $\overline{sg}(x) = 1 - sg(x)$.

1. *Remainder* $Rem(x, y)$ "the remainder of the divisione of y by x":
   (a) $Rem(x, 0) = 0$
   (b) $Rem(x, y + 1) = S(Rem(x, y) \cdot sg(|y - S(Rem(x, y))|)$
2. *Quotient* $qt(x, y)$ ="quotient of the divisione of y by x":
   (a) $qt(x, 0) = 0$
   (b) $qt(x, y + 1) = qt(x, y) + \overline{sg}(|x - S(Rem(x, y))|)$
3. *Limited sum*:
   (a) $\sum_{y \leq 0} f(x, y) = f(x, 0)$
   (b) $\sum_{y \leq z+1} f(x, y) = \sum_{y \leq z} f(x, y) + f(x, z + 1)$
4. *Limited product* : $\Pi_{y \leq x} f(x, y)$ (analogous).

A relation is primitive recursive, if its characteristic function is. Let $\chi(\phi) = 1$ iff $\phi$ is true. Some examples of such relations are the following:

1. *Characteristic function of equality*: $\chi_=(x, y) = \overline{sg}(|x - y|)$
2. *Characteristic function of the relation* $<$: $\chi_<(x, y) = sg(y \dot{-} x)$
3. The relations obtained from the primitive recursive relations by means of connectives and *bounded quantifiers* in turn primitives recursive:
   (a) *Propositional connectives*: $\chi(\neg\phi) = 1 - \chi(\phi)$, $\chi(\phi \wedge \psi) = \chi(\phi) \cdot \chi(\psi)$.
   (b) *Bounded quantifiers*: $\chi(\forall y \leq x \theta(x, y)) = \Pi_{y \leq x} \chi(\theta)(x, y)$
   (c) *Exercise.* Define the characteristic function of $\vee$ and of the bounded existential quantifier.

4. *Divisibility* ("$x$ divides $y$"): $x|y \leftrightarrow \exists z \leq y(x \cdot z = y)$
5. *Prime numbers* ("$x$ is a prime number"):

$$x \geq 2 \wedge \forall y \leq x(y|x \rightarrow y = 1 \vee y = x)$$

6. *Operator of bounded minimization* ("the minumum $y$ less or equal to $x$ such that..."):

$$\mu y \leq z.R(x,y) = \begin{cases} min\ y.R(x,y) & \text{if } \exists y \leq z R(x,y) \\ 0 & \text{otherwise} \end{cases}$$

where $R(x,y)$ is primitive recursive.
Observe that: $\mu y \leq z.R(x,y) = \sum_{y \leq z}(y \cdot g(x,y))$, where:

$$g(x,y) = \begin{cases} 1 & \text{if } R(x,y) \wedge \forall z < y \neg R(x,z) \\ 0 & \text{otherwise} \end{cases}$$

7. *Definition by cases.* Let $g_0, g_1, h$ be primitive recursive. Then also the following is primitive recursive:

$$f(x) = \begin{cases} g_0(x) & \text{if } h(x) = 0 \\ g_1(x) & \text{otherwise} \end{cases}$$

Just take $f(x) = g_0(x) \cdot \overline{sg}(h(x)) + g_1(x) \cdot sg(h(x))$.
8. *The sequence of prime numbers*, where $p(x) = p_x =$ the $x - th$ prime in increasing order:
   (a) $p_0 = 2$
   (b) $p_{x+1} = \mu y \leq p_x! + 1($"$y\ prime \wedge y > p_x$"$)$

The bound to the minimization operator is essential, and is obtained from the proof of Euclid's theorem on the infinity of the prime numbers.

To encode finite sequences of numbers *by single numbers*, we will apply this well-known result:

> (*Fundamental theorem of arithmetic* or unique prime-factorization theorem). Every integer greater than 1 either is prime itself or is the product of prime numbers, and that this product is unique, up to the order of the factors.

There are actually many methods to encode the finite sequences in primitive recursive way; a possible coding is as follows (see Odifreddi (1989-1999), Vol. I, pp. 88-90) and exploits the notions and results just introduced:

1. *Sequence.* $x = \langle x_1, ..., x_n \rangle = p_0^n \cdot p_1^{x_1} \cdot ... \cdot p_n^{x_n}$. Note that the first exponent n, gives the length of the sequence.
2. *Projection.* $(x)_i = exp(x, p_i) = \mu v \leq x(p_i^v | x \wedge \neg(p_i^{v+1}|x))$
3. *Length.* $lh(x) = (x)_0$
4. $Seq(x)$="is a sequence". $Seq(x) \leftrightarrow \forall i \leq x(i > 0 \wedge (x)_i \neq 0 \rightarrow i \leq lh(x))$
5. *Concatenation.* If $Seq(x)$ and $Seq(y)$, then: $x*y = p_0^{lh(x)+lh(y)} \cdot \Pi_{i<lh(x)} p_{i+1}^{(x)_{i+1}} \cdot \Pi_{i<lh(y)} p_{lh(x)+i+1}^{(y)_{i+1}}$. Otherwise $x * y = 0$.

In a similar way we can encode the notion of initial segment $x$ of a sequence $y$ (*Exercise*). Notice that $Seq(x)$ says that the code of a finite sequence has the form:

$$x = p_0^n \cdot p_1^{x_1} \cdot ... \cdot p_n^{x_n} \cdot p_{n+1}^0 \cdot p_{n+2}^0 \cdot p_{n+3}^0 \cdot ...$$

where $n = lh(x)$ says that the sequence ends at $x_n$ and all non-zero exponents (i.e. the elements of the sequence) occur among $x_1, ..., x_n$.

We now have the tools to prove (Péter) the closure of the set of primitive recursive functions under *recursion on the course of the values.*

**Theorem 7.** *The following schema of recursion on the course of the values is primitive recursive:*

1. $F(0, y) = g(y)$
2. $F(x+1, y) = h(\overrightarrow{F}(x, y), x, y)$

*Proof.* Let us show that the history of $F$, namely $\overrightarrow{F}(x, y) = \langle F(0, y), ..., F(x, y)\rangle$, is primitive recursive, for $g, h$ primitive recursive. It follows that also the recursion on the course of values is primitive recursive: In fact just define $\overrightarrow{F}$ as follows:

1. $\overrightarrow{F}(0, y) = \langle g(y)\rangle$
2. $\overrightarrow{F}(x+1, y) = \overrightarrow{F}(x, y) * < h(\overrightarrow{F}(x, y), x, y) >$

Therefore $F(x, y) = (\overrightarrow{F}(x, y))_{x+1}$.

QED

Recall that the primitive recursive functions are total. Not only, but we can say more about their totality.

**Definition 6.** *Let $f(x_0, ..., x_k)$ be a total function and $\mathsf{T}$ an extension of Robinson arithmetic $\mathsf{Q}$. We say that $f(x_0, ..., x_k)$ is provably total in $\mathsf{T}$, if there is a formula $\psi(x_0, ..., x_k, y)$ such that:*

1. *$\psi(\overline{n_0}, ..., \overline{n_k}, \overline{f(n_0, ..., n_k)})$ is true, for all numbers $n_0, ..., n_k$ (where $\overline{n}$ is the numeral $\overbrace{SSS...S}^{n-times}0$ denoting $n$).*
2. *$\mathsf{T} \vdash \forall x_0 ... \forall x_k \forall y \forall z(\psi(x_0, ..., x_k, y) \wedge \psi(x_0, ..., x_k, z) \rightarrow y = z)$.*
3. *$\mathsf{T} \vdash \forall x_0 ... \forall x_k \exists y \psi(x_0, ..., x_k, y)$*

**Theorem 8.** (Parsons 1970) *Let $I\Sigma^0_1$ be the subtheory of Peano Arithmetic, obtained by restricting the induction schema to the $\Sigma^0_1$ formulas. Hence $f(x_0, ..., x_k)$ is primitive recursive if and only iff it is provably total in $I\Sigma^0_1$.*

*Proof.* In Parsons (1970). See also Buss (1998), where this theorem is proved using the "witnessing" method, which we will illustrate in ch.7.4. QED

We have seen that not all computable total functions are primitive recursive. However, Ackermann's function, for instance, can be obtained by further expanding the class of recursive functions with the *regular minimization scheme*:

$$f(x) = \mu y.(g(x, y) = 0)$$

as long as $g$ is total computable and $\forall x \exists y(g(x, y) = 0)$. Note that without this clause we go out from total functions. We calculate $g(x, 0), g(x, 1), g(x, 2)$... If a value y exists for which $g(x, y) = 0$, the computation sooner or later ends; but if we do not put this clause, the computation might never end. This is another way of incorporating induction, the minimum principle being equivalent to complete induction. Gödel and Kleene actually showed that this class, the total recursive functions, can be characterized also without the primitive recursion.

**Theorem 9.** *The class of total recursive functions is the smallest class containing sum, product, projections, the characteristic function of equality and is closed under composition and under $\mu$-recursion.*

We have so exhausted the notion of computability? Not really. We aim at a formalization of computable functions such that:

1. we can give an effective list $f_0, f_1, f_2, ...$ of their programs, containing all and only the programs of the computable functions

2. Such a list must be *uniformly effective*, in the sense that there is an algorithm $\Phi$ such that for all $e, n$:

$$\Phi(e,n) = f_e(n)$$

Now we will see that this cannot be done, if we limit ourselves to *total* functions, because of the phenomenon of *diagonalization*: for total recursive functions there is no *universal function* $\Phi$ as above. This method is applicable to any case where the sets of instructions can be effectively listed. Suppose that the above claims are satisfied by the total computable functions and take $h(x) = \Phi(x,x)+1$. Also $h$ is total, hence corresponds to some $f_e$ in the list. But then we have a contradiction, since $f_e(e) = h(e) = \Phi(e,e)+1 = f_e(e)+1$. Hence $h$ cannot be a member of the above list, which cannot be therefore exaustive. We must therefore abandon one of the above conditions. The problem does not arise if, while maintaining conditions 1. and 2., we admit partial functions. Let us admit therefore partial functions, i.e. that are not defined on some numbers. With the notation $\phi(x)\downarrow$ we mean that the function is defined on $x$, while the notation $\phi(x)\uparrow$ mean that the function is not defined at $x$. With writing $\phi(x) \simeq \psi(x)$ we understand now that either $\phi(x)\uparrow$ and $\psi(x)\uparrow$, or $\phi(x)\downarrow$ and $\psi(x)\downarrow$ and $\phi(x) = \psi(x)$. The minimization scheme is now as follows. If $\phi(x,y)$ is partial recursive:

$$\psi(x) \simeq \mu y(\phi(x,y) \simeq 0 \wedge \forall z \leq y\phi(x,z)\downarrow)$$

It is undefined if there is no such $y$. Adding this scheme to the primitive recursive functions (where equality has to be intended between partial functions) we have the model of Kleene's $\mu-$recursive functions[1].

The above definition can also be reformulated as follows: the partial recursive functions can be obtained by adding this minimization scheme, rather than what we added before:

$$\phi(x_0,...,x_n) \simeq \mu y.R(x_0,...,x_n,y)$$

where $R(x_0,...,x_n,y)$ is a recursive relation and $\phi(x_0,...,x_n)\uparrow$ if does not exist y satisfying $R(x_0,...,x_n,y)$. The partial recursive functions are not closed with respect to this scheme, if $R(x_0,...,x_n,y)$ is only recursively enumerable. Actually, if $R(x,y)$ is recursive, we can define

$$\phi(x_0,...,x_n) \simeq \mu y.((\overline{sg}\chi_R(x_0,...,x_n,y)) \simeq 0)$$

**Theorem 10.** (Kleene 1936) *Let $n > 0$; there are primitive recursive relations $T^n(x,x_0,...,x_{n-1},z)$ and $U(z)$ such that, for each recursive n-ary partial function $\psi$, there exists $e \in \mathbb{N}$ such that:*

$$\psi(x_0,...,x_{n-1}) \simeq U(\mu z.T^n(e,x_0,...,x_{n-1},z))$$

$$\psi(x_0,...,x_{n-1})\downarrow \text{ iff } \exists z(T^n(e,x_0,...,x_{n-1},z))$$

*The relation $T^n(e,x_0,...,x_{n-1},z)$ must be read as: "z encodes a computation of the function whose code is e on input $x_0,...,x_{n-1}$".*

*Proof.* The proof is very laborious and we refer to Odifreddi (1989-1999) pp. 90-6 for a detailed account. The basic idea is as follows: first of all, we associate numbers to the initial functions and to all scheme generating the recursive functions, encoding them *à la* Gödel. Then we define a computation tree, whose nodes are labelled by triples $\langle e, \langle x_0,...,x_n\rangle, b\rangle$ where $e$ is the code of a function $f$, the sequence $\langle x_0,...,x_n\rangle$ represents its argument and $b$ is the value of $f(x_0,...,x_n)$. Internal nodes tell us inductively through which scheme, applied to the function that labels the children of a node, we arrived at the function that labels that node. Leaves correspond to initial functions and the root is labelled by the triple corresponding to the function whose computation

[1] In Odifreddi (1989-1999) p. 128 is made clear that the set of partial recursive functions *is not closed* under the simplified scheme $\phi(x) \simeq \mu y.(\psi(x,y) \simeq 0)$.

tree we are defining. The whole tree will be encoded like this: suppose that the node $k$ has $n+1$ children, which are the roots of subtrees $T_0, ..., T_n$; then the tree generated by $k$ has code $z = \langle k, \ulcorner T_0 \urcorner, ..., \ulcorner T_n \urcorner \rangle$. Finally, we define $T^n(e, x_0, ..., x_{n-1}, z)$ iff $e$ encodes $\psi(x_0, ..., x_{n-1})$ and $z$ encodes the computation tree whose root is labelled by $k = \langle \ulcorner \psi \urcorner, \langle x_0, ..., x_{n-1} \rangle, b \rangle$. With the use of the projections also define $U(z) = b$. Indeed, note that $(z)_1 = k$; but $k = \langle \ulcorner \psi \urcorner, x_0, ..., x_{n-1}, u \rangle$, hence $((z)_1)_1 = \ulcorner \psi \urcorner$, $((z)_1)_2 = \langle x_0, ..., x_{n-1} \rangle$ and so on. To translate this representation into a primitive recursive predicate that says that a given number encodes a computation tree, we in fact only need primitive recursive predicates and functions, mainly codes of sequences and projections. QED

We have in fact obtained the *enumeration theorem*: for each $n$, there exists a *universal function* concerning the class of partial recursive functions in $n$ variables:

$$K^n(e, x_0, ..., x_{n-1}) \simeq U(\mu z T^n(e, x_0, ..., x_{n-1}, z))$$

Moreover, if we define $\phi_e^n(x_0, ..., x_{n-1}) \simeq K^n(e, x_0, ..., x_{n-1})$, then for all $e$ each $\phi_e^n$ is partial recursive and each $n-ary$ partial recursive function correspond to some $\phi_e^n$ in the list. But we can also omit the reference to $n$-arity and see from the above that there is an *universal function* $K(u, x)$ that generates all partial functions, that is, such that, for each arity and for each partial recursive function $\phi(x_0, ..., x_{n-1})$, exists a code $e$ for which we have:

$$\phi(x_0, ..., x_{n-1}) \simeq K(e, \langle x_0, ..., x_{n-1} \rangle)$$

Observes that $T^n(e, x_0, ..., x_{n-1}, y)$ implies $((y)_1)_1 = e$ and $((y)_1)_2 = \langle x_0, ..., x_{n-1} \rangle$. Then we place:

$$K(e, x) \simeq U(\mu y.T(y) \wedge ((y)_1)_1 = e \wedge ((y)_1)_2 = x)$$

where $T(y)$ is the primitive recursive predicate formalizing “ y is a computation tree ”, according to the previous construction. It has thus:

$$K(e, \langle x_0, ..., x_{n-1} \rangle) \simeq U(\mu y.T(y) \wedge ((y)_1)_1 = e \wedge ((y)_1)_2 = \langle x_0, ..., x_{n-1} \rangle)$$
$$\simeq \phi^n(x_0, ..., x_{n-1})$$

We write $\phi_e^n(x_0, ..., x_{n-1})$ for $K(e, \langle x_0, ..., x_{n-1} \rangle)$.

A universal Turing machine is essentially a machine that computes a similar function. It is also concluded from the above that there are exactly $\aleph_0$ *partial* recursive functions and exactly $\aleph_0$ *total* recursive functions: by the Church's thesis constant functions $c_n(x) \simeq n$ are total recursive and therefore there are at least $\aleph_0$ *total* recursive functions. Moreover, they cannot be more than $\aleph_0$, by the normal form theorem. It is also known that, being the set of functions (computable or not) from $\mathbb{N}$ to $\mathbb{N}$ of cardinality $2^{\aleph_0}$. This means that the computable functions are a small minority. We must make a few remarks about the *indices*. First of all each recursive partial function has infinite indices; for instance:

$$\psi_e(x) \simeq \psi_e(x) + Z(P_0^m(x))$$

And yet the indices of the two functions are different. The index for $\psi_e(x) + Z(P_0^m(x))$ is given by the sequence code $\langle 3, e, \langle 3, \langle 2, m, 1 \rangle, \langle 0 \rangle \rangle, e^+ \rangle$, where $e^+$ is the index of $+$.

**Theorem 11.** ("Padding lemma") *There is a total function $f(e, m, w)$ such that $\phi_e(x) \simeq \phi_{f(e,m,n)}(x)$, where $f(e, m, 0) < f(e, m, 1) < f(e, m, 2) < ...$ are all indexes of the same function.*

*Proof.* If $z = \langle 3, \langle 2, m, 1 \rangle, \langle 0 \rangle \rangle = \ulcorner Z(P_0^m(x)) \urcorner$,

let $g(e, m) = \langle 3, e, z, e^+ \rangle = \ulcorner \psi_e(x) + Z(P_0^m(x)) \urcorner$; observe that $\phi_e(x) \simeq \phi_{g(e,m)}(x)$. Let therefore $f(e, m, 0) = e$ and $f(e, m, n+1) = g(f(e, m, n), m)$.

QED

**Theorem 12.** (Theorem of parameters, or *s-m-n*-theorem) *Given $m, n \in \mathbb{N}$ there exists a function $s(e, x_0, ..., x_{n-1})$ such that:*

$$\phi_e(x_0, ..., x_{n-1}, w_0, ..., w_{m-1}) \simeq \phi_{s(e, x_0, ..., x_{n-1})}(w_0, ..., w_{m-1})$$

*Proof.* We want constant functions $\phi_{f(0)}(w_0, ..., w_{m-1}) = 0$, $\phi_{f(1)}(w_0, ..., w_{m-1}) = 1$ etc. Thus, let us define:

1. $f(0) = \langle 3, \langle 2, m, 1\rangle, \langle 0\rangle\rangle$
2. $f(n+1) = \langle 3, f(n), \langle 1\rangle\rangle$

Then observe that:

1. $\phi_{f(0)}(w_0, ..., w_{m-1}) = Z(P^m_1(w_0, ..., w_{m-1}))$
2. $\phi_{f(n)}(w_0, ..., w_{m-1}) = \overbrace{SSS...S}^{n-times}(Z(P^m_1(w_0, ..., w_{m-1})) = n$

Hence, abbreviating $w = w_0, ..., w_{m-1}$ and $x = x_0, ..., x_{n-1}$:

$$\phi_e(x, w) \simeq \phi_e(\phi_{f(x_0)}(w), ..., \phi_{f(x_{n-1})}(w), P^m_1(w), ..., P^m_m(w))$$

Let therefore:

$$s(e, x_0, ..., x_{n-1}) = \langle 3, f(x_0), ..., f(x_{n-1}), \langle 2, m, 0\rangle, ..., \langle 2, m, m-1\rangle, e\rangle$$

And therefore we have $\phi_e(x, w) \simeq \phi_{s(e,x)}(w)$ QED

Many results that have been proved so far (enumeration theorem, *s-m-n* theorem, padding lemma) mention indices in their statements or in their proofs: do these results depend on the particular way in which indices were defined? We call acceptable a system that resemble the standard one, i.e. if it is possible to go effectively from the one to the other. More precisely, if $\psi_e$ is the standard one, then a system of indices $\zeta_e$ is acceptable if there are computable functions $g, f$ such that $\psi_e = \zeta_{f(e)}$ and $\psi_{g(e)} = \zeta_e$. In this respect we have this result:

**Theorem 13.** *An enumeration of the partial recursive functions $\{\phi_e\}_{e\in\mathbb{N}}$ is an accettable indexing if and only if:*

1. *for every partial recursive function $\psi$ exists an index $e$ such that $\psi \simeq \phi_e$.*
2. *There is a universal function.*
3. *The parameter theorem is fulfilled..*

It follows that every acceptable system of indices satisfies the *Recursion Theorem.*

**Theorem 14.** (Recursion Theorem) *Let $f$ be total recursive; then there exists $n \in \mathbb{N}$ such that $\phi_n(x) \simeq \phi_{f(n)}(x)$, namely $n$ and $f(n)$ compute the same function: we say that $n$ is a fixed point of $f$.*

*Proof.* Let $u \in \mathbb{N}$ and let $\phi(x)$ be defined by the following instructions: apply the set of instructions $P_u$ coded by $u$, to the input $u$; if the computation terminates and gives output $w$, then take the set $P_w$ of the instructions encoded by $w$ and apply it to $x$. If $P_w$ applied to $x$ halts and returns output $z$, put $\phi(x) \simeq z$. More formally, whereas the instructions to $\phi$ depends on $u$, let $g$ the function that, given $u$, returns the code for these instructions:

$$\phi_{g(u)}(x) \simeq \begin{cases} \phi_{\phi_u(u)}(x) & \text{if } \phi_u(u)\downarrow \\ \uparrow & \text{otherwise} \end{cases}$$

Let $\phi_e(x) = f(g(x))$ and $n = g(e)$; then $\phi_n \simeq \phi_{g(e)} \simeq \phi_{\phi_e(e)} \simeq \phi_{f(g(e))} \simeq \phi_{f(n)}$ QED

*Example.* There is an index $e$ such that for all $n$, $\phi_e(n) \simeq e$, that is, this function prints its name. Let $f(x, y) = x$; by the $s-m-n$ theorem there exists a function $g$ such that $f(x, y) \simeq \phi_{g(x)}(y) = x$. For the theorem of fixed point exists $n$ such that $\phi_{g(n)}(x) \simeq \phi_n(x)$. Thus:

$$n = f(n, y) \simeq \phi_{g(n)}(y) \simeq \phi_n(y)$$

*Example.* We can use the theorem of fixed point and the $s-m-n$ theorem to prove the Ackermann function. Let $K(e, x)$ a universal function and define by cases:

1. $g(n, 0, y) \simeq y + 1$
2. $g(n, x + 1, 0) \simeq K(n, \langle x, 1\rangle)$
3. $g(n, x + 1, y + 1) \simeq K(n, \langle x, K(n, \langle x + 1, y\rangle)\rangle)$

For the fixed point and for s-m-n, we find an $n$ such that:

$$g(n, x, y) \simeq \phi_{h(n)}(x, y) \simeq \phi_n(x, y)$$

Thus:

1. $\phi_n(0, y) \simeq \phi_{h(n)}(0, y) \simeq g(n, 0, y) = y + 1$
2. $\phi_n(x + 1, 0) \simeq \phi_{h(n)}(x + 1, 0) \simeq g(n, x + 1, 0) \simeq K(n, \langle x, 1\rangle) \simeq \phi_n(x, 1)$
3. $\phi_n(x + 1, y + 1) \simeq \phi_{h(n)}(x + 1, y + 1) \simeq g(n, x + 1, y + 1) \simeq$
   $\simeq K(n, \langle x, K(n, \langle x + 1, y\rangle)\rangle) \simeq \phi_n(x, \phi_n(x + 1, y))$

As to the first example, functions that prints their own code as in that case are called *quines*. A *quine* is an easy example self-replicating computer program which takes no input and produces a copy of its own source code as its only output. The name "quine" was coined by Douglas Hofstadter, in his popular science book Hofstadter (1979), in honor of the philosopher Willard Van Orman Quine (1908-2000). There is a famous anecdote that when Descartes presented Christina of Sweden with the hypothesis that animals constitute a form of mechanical automaton, she pointed to a clock and exclaimed: "Let's see if it produces a child". Organisms, unlike machines, are self-organising systems that self- reproduce. The view of animals as machines (la *bête machine*), in other words, ran into serious difficulties in the face of the objection that living organisms, in general, unlike machines (as has long been assumed), have the capacity to reproduce themselves. Speaking more abstractly, self-replication is any behaviour of a dynamic system that leads to the construction of a copy of itself. Since the time of Descartes and La Mettrie, many things (starting with the concept of "machine" itself) have changed, but the opposition between vitalistic, organicistic and mechanistic conceptions of life is still strong. Certain mechanistic tenets conceive of living organisms as very complex machines programmed by genetic software, just as, conversely, attempts are made to describe living organisms in terms of self-replicating automatons. Anti-mechanistic conceptions instead emphasise the irreducible diversity between organisms and machines. The fixed point theorem and the diagonalization technique shed light on this problem (see also Rogers (1987) pp. 188-90 and Odifreddi (1989-1999) pp. 170-74 for a more in-depth discussion) and thanks to them it can be demonstrated that there must be a machine that, regardless of the input, constructs its own replica. The problem of self-replication of machines began to become a concrete and "engineering" problem with Von Neumann, who formalized the idea of cellular automata in order to create a theoretical model for a self-reproducing machine. He drew a general outline of his self-replicating automaton that anticipated some concepts of current cell biology, such as those of translation and transcription (the two fundamental stages of the protein synthesis process). In fact, the insights behind Von Neumann's original model anticipated some of Watson and Crick's (1953) discoveries concerning the functioning of DNA. At the Hixon Symposium in Pasadena, California on September 1948 he compared the functions of genes to self-reproducing automata; its universal constructor is a self-replicating machine in a cellular automata (CA) environment. Cellular automata are mathematical models he used to simulate complex systems representing real-world phenomena studied in physics and biology. In this research he made use of some concepts introduced by Turing, for example the Universal constructor, that is, able to build any machine starting from a its description, and in particular to self-reproduce, a concept inspired by the Universal Turing Machine (see Von Neumann (1951)).

### 2.2. Recursively enumerable sets

Informally, a set is computably enumerable (o recursively enumerable), if there is a computable procedure to list its elements.

**Definition 7.** *A set $X \subseteq \mathbb{N}$ is recursively enumerable (or computably enumerable) iff either $X = \emptyset$, or $X = Range(f) = \{f(0), f(1), f(2), ...\}$, for some $f$ total recursive.*

Note that if $X$ is recursive, then it is also recursively enumerable; in case of $X = \emptyset$, it is obvious; in case of $X$ *finite*, e.g. $X = \{k_0, ..., k_n\}$, take $f$ total recursive defined as follows:

$$f(x) \simeq \begin{cases} k_x & \text{if } x \leq n \\ k_n & \text{otherwise} \end{cases}$$

It is clear that $X$ is the range of $f$. If $X$ is *infinite* and $\chi_X$ is the characteristic function:

1. $f(0) \simeq \mu y.(\chi_X(y) = 0)$
2. $f(n+1) \simeq \mu y.(\chi_X(y) = 0 \wedge f(n) < y)$

Note that $Range(f) = X$.

**Lemma 1.** *The following are equivalent:*

1. *Either $A = \emptyset$ or $A = Range(f)$ for some $f$ total recursive.*
2. *$A = Dom(\psi)$, for $\psi$ partial recursive.*
3. *$A = Range(\phi)$, for $\phi$ partial recursive.*

*Proof.* 1. $\Rightarrow$ 2. if $A = \emptyset$, then $A = Dom(\phi)$, where $\phi$ is the function everywhere divergent. If $A \neq \emptyset$, $A = Range(f)$, then define $\psi$ using the following instructions:

1. generate $Range(f)$
2. when $y$ appears, put $\psi(y) \simeq y$

So we get $Dom(\psi) = Range(f) = A$.

2. $\Rightarrow$ 3. Let $A = Dom(\psi)$, let moreover $\phi(x) \simeq x + 0 \cdot \psi(x)$; note that $A = Range(\phi)$.

3. $\Rightarrow$ 1. Let $A = Range(\phi)$, let therefore:

$$f(\langle x, t\rangle) \simeq \begin{cases} z & \text{if } \phi(x) \downarrow \text{ in at most } t \text{ steps with output } z \\ a \in A & \text{otherwise} \end{cases}$$

Then $A = Range(f)$. QED

**Remark 2.** *Note that if we had omitted the bound $t$ we would not have guaranteed the total character of $f$; the reason is that we can not recursively decide whether $\phi(x)\downarrow$ ( unsolvability of the "halting problem"), but on the contrary we can decide whether $\phi(x)\downarrow$ in at most $t$ steps. Moreover, in 1. the function $f$ can be taken primitive recursive: suppose that $A \neq \emptyset$; if $f(x) = \phi_e(x) = y$, then $\exists s T(e, x, s) \wedge U(s) = y$. So, define $g(z) = U((z)_1)$, if $T(e, (z)_0, (z)_1)$, and an element $a \in A$ otherwise. Show that the range of $g$ is $A$.*

There is another interesting characterization of the computably enumerable sets which will come in handy in sections 2.4 and 8.3.

**Theorem 15.** *The following are equivalent:*

1. *$A$ is computably enumerable.*
2. *There is a total computable binary function $f(x, s)$ (i.e. with values $0, 1$) such that for every $x$, $f(x, 0) = 0$, there is at most one $s$ such that $f(x, s+1) \neq f(x, s)$, and $\lim_s f(x, s) = \chi_A(x)$ =characteristic function of $A$.*
3. *There is a computable sequence of finite sets $A_s, s \in \mathbb{N}$, such that for all $s, A_s \subseteq A_{s+1}$ , and $A = \cup_s A_s$ .*

**Lemma 2.** *$X$ is recursive iff $X$ and $\overline{X}$ are both computably enumerable*

*Proof.* ⇒ Let $\chi_X(x)$ be te characteristic funcion of $X$ (by hypothesis computable) and let us consider the following:

$$\phi(x) \simeq \begin{cases} 0 & \text{if } \chi_X(x) = 0 \\ \uparrow & \text{otherwise} \end{cases}$$

$$\psi(x) \simeq \begin{cases} 1 & \text{if } \chi_X(x) = 1 \\ \uparrow & \text{otherwise} \end{cases}$$

and observe that $X = Dom(\psi)$ and $\overline{X} = Dom(\phi)$.

⇐ If $X$ and $\overline{X}$ are computably enumerable not empty, are then generated respectively by $f$, $g$ total recursive, i.e. $X = \{f(0), f(1), f(2)...\}$, $\overline{X} = \{g(0), g(1), g(2), ...\}$. To find out whether a number belongs to $X$ just generate both sets. QED

It is worth pointing out the difference with this result:

**Lemma 3.** *The following statements are equivalent:*

1. *$X$ is recursive.*
2. *Either $X = \emptyset$ or $X$ is the range of a non decreasing recursive function $f$ (i.e. if $a > b$ then $f(b) \geq f(a)$)*

*Proof.* ⇒ Let $X \neq \emptyset$ recursive; let $a$ its smallest element and let $f(0) = a$,

$$f(n+1) \simeq \begin{cases} n+1 & \text{if } n+1 \in X \\ f(n) & \text{otherwise} \end{cases}$$

Note that $f$ is non-decreasing.

⇐ Let $X$ infinite and range of $f$ non decreasing recursive; to know if $x \in X$, the test is as follows: Search the minimum $n$ such that $f(n) > x$. We will have that $x \in X$ iff $x \in \{f(0), ..., f(n)\}$ QED

**Definition 8.** *With notation $W_e$ we mean the computably enumerable set $Dom(\phi_e)$. Hence every computably enumerable set can be written in this form.*

Moreover, from the normal form theorem:

$$W_e = \{\langle x_0, ..., x_{n-1}\rangle | \exists y T(e, x_0, ..., x_{n-1}, y)\}$$

It follows that $X$ is computably enumerable iff there exist a recursive relation $R$ such that:

$$X = \{\langle x_0, ..., x_{n-1}\rangle | \exists v R(x_0, ..., x_{n-1}, v)\}$$

A direction follows from the above; for the other direction, suppose:

$$X = \{\langle x_0, ..., x_{n-1}\rangle | \exists v R(x_0, ..., x_{n-1}, v)\}$$

Then we take $\psi(x_0, ..., x_{n-1}) \simeq \mu z.R(x_0, ..., x_{n-1}, z)$. Note that $X = Dom(\psi)$.

We now propose again, in this different context, the fundamental result about the unsolvability of the halting problem, already highlighted in the chapter on Turing machines.

**Theorem 16.** *The problem "$x \in W_x$" is undecidable.*

*Proof.* Let us suppose that the following function is computable:

$$f(x) = \begin{cases} 1 & \text{if } \phi_x(x) \downarrow \\ 0 & \phi_x(x) \uparrow \end{cases}$$

Then take:

$$g(x) = \begin{cases} 0 & \text{if } f(x) = 0 \\ \uparrow & f(x) = 1 \end{cases}$$

Observe that $g$ is partial recursive; let $g = \phi_e$, for some $e$. Hence $e \in W_e$ iff $\phi_e(e) \downarrow$ iff $g(e) = 0$ iff $e \notin W_e$ (contradiction). Hence $f$ cannot be computable. It follows that also the following is incomputable:

$$h(x, y) = \begin{cases} 1 & \text{if } \phi_x(y) \downarrow \\ 0 & \text{otherwise} \end{cases}$$

If it were, then it would also be $f(x) = h(x, x)$. Hence also the problem "$y \in W_x$" is unsolvable. QED

**Corollary 1.** *The following ("diagonal set" or "halting set") is computably enumerable but not recursive:*

$$K = \{n \in \mathbb{N} | \phi_n(n) \downarrow\} = \{n \in \mathbb{N} | n \in W_n\}$$

*Proof.* Let indeed $\psi(x) = \phi_x(x)$, and note that it converges, namely is defined, on $x$ iff $x \in K$, i.e. $x \in Dom(\psi)$ iff $x \in K$; hence $K = Dom(\psi)$ and therefore K is computably enumerable But is not recursive: indeed its complement $\overline{K}$ is not computably enumerable Note that $x \in \overline{K}$ iff $\phi_x(x) \uparrow$ iff $x \notin W_x$. Since each computably enumerable set is of the form $W_x$, this means that $\overline{K}$ differs from each of them in at least one element. Hence is different from all computably enumerable sets. QED

*Example.* There is no effective way of deciding, given $x$ if $\phi_x$ is total. It is observed that the set $Tot = \{x | \phi_x \ totale\}$ is not computably enumerable For if it were, there would be some $f$ recursive of which would be the range:

$$\phi_{f(0)}, \phi_{f(1)}, \phi_{f(2)}, \ldots.$$

But then we could take $g(x) = \phi_{f(x)}(x) + 1$ noting that $g \neq \phi_{f(x)}$, for all $x$. For if we had $g = \phi_{f(e)}$, for some e, then $g(e) = \phi_{f(e)}(e) = \phi_{f(e)}(e) + 1$

*Index set.* Let $\mathcal{F}$ a set of partial recursive functions; the set $\mathcal{F}^*$ of all the indices of the functions in $\mathcal{F}$, assuming that if $x \in \mathcal{F}^*$ and $\phi_x \simeq \phi_y$ then $y \in \mathcal{F}^*$, is called "index set". Note that the set $Tot$ of the previous example is an index set, but $K$ is not; take indeed $f(n) =$ index of $\{n\}$. For the fixed point theorem there exists $e$ such that $W_{f(e)} = W_e$, whereby $e \in W_e$ iff $e \in W_{f(e)}$ iff $e \in \{e\}$. If now we take a different index $j$ of $\{e\}$, then $j \notin W_j = \{e\}$, i.e. $\phi_j(j) \uparrow$ and therefore $j \notin K$. Note that an index set $\mathcal{F}$ contains all possible "programs" to calculate the functions contained in $\mathcal{F}$.

**Theorem 17.** (Rice theorem) *Each nontrivial property of programs is undecidable: if $\mathcal{F}$ is a class of partial recursive functions and $\mathcal{F}^*$ is its index set, then $\mathcal{F}^*$ is recursive iff either is empty, or coincides with the class of all partial recursive functions.*

*Proof.* Reduces $K$ to $\mathcal{F}^*$; we exclude cases in which $\mathcal{F}^* = \emptyset$ and $\mathcal{F}^* = \mathbb{N}$; we exclude also the case everywhere undefined function $\phi_\emptyset$. Let therefore $a \in \mathcal{F}^*$ and let:

$$\psi(x, y) = \begin{cases} \phi_a(y) & \text{if } x \in K \\ \uparrow & \text{otherwise} \end{cases}$$

by parametrization we obtain $\psi(x, y) \simeq \phi_{h(x)}(y)$. Then:

1. If $x \in K$, then $\psi(x, y) = \phi_a(y)$, from which $\phi_a \simeq \phi_{h(x)}$ and $h(x) \in \mathcal{F}^*$.
2. If $x \notin K$, then $\psi(x, y) \uparrow$, from which $\phi_\emptyset \simeq \phi_{h(x)}$ and $h(x) \notin \mathcal{F}^*$

If therefore $\mathcal{F}^*$ would be recursive, also K would be. QED

An *index-set* is intended to code properties of functions, not of programs, i.e. not depending on the particular algorithm used for computing that functions; but Rice's theorem says that all nontrivial properties of index-sets are undecidable.

Lastly, an important characterisation of computably enumerable sets is that connected to the negative solution of Hilbert's $X$ problem. A Diophantine equation is a polynomial with integer coefficients for which we seek integer solutions. In general they have the form: $f(x_0, ..., x_m) = 0$, or $f(x_0, ..., x_m) = g(x_0, ..., x_m)$ where $f, g$ are polynomials with integer coefficients.

### 2.3. Hilbert's tenth problem

The tenth on the list of mathematical problems that Hilbert posed in 1900 asks: "given a Diophantine equation with any number of unknowns and integer coefficients, define a process that could end in a finite number of operations, to determine whether the equation has integers solutions" . We consider *families* of Diophantine equations $D(a_0, ..., a_k, x_0, ..., x_m) = 0$, where:

1. the variables $a_0, ..., a_k$ are called *parameters*.
2. the variables $x_0, ..., x_m$ are called *unknowns*.

By setting the parameters we obtain an equation; on the basis of the parameters, the equation may or may not have solutions for the unknowns (that is to say values which replaced the unknowns make it true). A set of the form:

$$X = \{\langle a_0, ..., a_k\rangle | \exists x_0 ... \exists x_m (D(a_0, ..., a_k, x_0, ..., x_m) = 0)\}$$

i.e. a set consisting of $k + 1$-tuples of *natural numbers* $\langle a_0, ..., a_k\rangle$ for which there exist solution vectors for the unknowns, $x_0, ..., x_m$, is called *Diophantine*. Indeed, the expression "integers solutions" , il Logic books, is sometimes replaced with "solutions in natural numbers". This is due to a Lagrange's theorem which states that every non-negative integer is the sum of four squares of integers. Considers, therefore $f(x_0, ..., x_m) = 0$ and suppose that we are looking for solutions in the naturals. Then we take $f(a_0^2+b_0^2+c_0^2+d_0^2, ..., a_m^2+b_m^2+c_m^2+d_m^2) = 0$. From Lagrange's four-square theorem, this is guaranteed to be possible. Substituting $a_0^2 + b_0^2 + c_0^2 + d_0^2, ..., a_m^2 + b_m^2 + c_m^2 + d_m^2$ in place of $x_0, ..., x_m$ into $f(x_0, ..., x_m)$, we obtain a Diophantine equation in $4m$ variables, all of which are integers.

**Theorem 18.** *The problem of determining the existence or nonexistence of solutions to a Diophantine equation in natural numbers is reducible to the problem of determining the existence or nonexistence of solutions to a Diophantine equation with integer values, and the opposite is also true.*

In 1934 Godel shows that the undecidable statement of his 1931 theorem can be expressed in the form: $Q_0x_0...Q_nx_n(f(x_0, ..., x_n) = 0)$ where $f$ is a polynomial with integer coefficients and $Q_0, ..., Q_n$ are quantifiers. subsequently Gödel shows that every statement of the form $\forall x\phi(x)$ with $\phi(x)$ primitive recursive is equivalent to one of the form:

$$\forall x_0...\forall x_m \exists y_0...\exists y_n(f(x_0, ..., x_m, y_0, ..., y_n) = 0)$$

In 1950 Martin Davis conjecture that a set is computably enumerable iff it is Diophantine. It also shows that every computably enumerable set $X$ has the form:

$$X = \{\langle a_0, ..., a_n\rangle | \exists z \forall y \leq z \exists x_0...\exists x_k f(a_0, ..., a_n, y, z, x_0, ..., x_k) = 0\}$$

It will take another two decades to eliminate the bounded quantifier "$\forall y \leq z$". In 1952 Julia Robinson shows that there is a polynomial $f$ such that $a^b = c$ if and only if $\exists z_0...\exists z_m(f(a, b, c, z_0, ..., z_m) = 0)$ under a sufficient condition that a Diophantine set of pairs $X = \{\langle u, v\rangle | \exists y_0, ..., \exists y_k g(u, v, y_0, ..., y_n) = 0\}$ exists, such that:

1. if $\langle u, v\rangle \in X$, then $u < v^v$
2. for all $k$ exists $\langle u, v\rangle \in X$ such that $u > v^k$.

If we call (JR) this hypothesis, the result is that if (JR) is true, then exponentiation is Diophantine. In 1959 Davis, Putnam and Robinson showed that each computably enumerable set is *exponential* Diophantine. Hence (JR) implies that every computably enumerable set is Diophantine. Remember that the exponential diophantine equations are $f(x_0, ..., x_m, y_0, ..., y_n) = 0$ where the $f$ is an exponential polynomial, i.e. constructed by addition, multiplication and exponentiation, e.g.

$$(x+1)^{y+2} + x^3 = y^{(x+1)^x} + y^4$$

Lastly, in 1970 Y. Matiyasevich find the required equation (JR) demonstrating that the Fibonacci sequence $F_0 = 1$, $F_1 = 1$ and $F_n = F_{n-1} + F_{n-2}$ is definable by a set of ten Diophantine equations and that the set $X = \{\langle u, v\rangle | u = F_{2v} \wedge v > 2\}$ satisfies (JR).

**Theorem 19.** (Matiyasevich 1970, Robinson 1950, Davis 1949, Putnam 1959) *Each computably enumerable set is Diophantine.*

Note that for the theorem of normal form $P(x_0, ..., x_k)$ is computably enumerable iff there exist a recursive relation $R(x_0, ..., x_k)$ such that:

$$P(x_0, ..., x_k) \leftrightarrow \exists y R(x_0, ..., x_k)$$

Hence also the converse of Matiyasevich theorem holds and it follows:

**Corollary 2.** *Computably enumerable sets coincide with the Diophantine sets.*

**Corollary 3.** *Hilbert's tenth problem has a negative solution.*

*Proof.* The halting set $K$ can be written in Diophantine form:

$$K = \{x | \exists y_0 ... \exists y_n (f(y_0, ..., y_n, x) = 0)\}$$

But $K$ is not computable, then there is no algorithm for deciding whether the equations:

$$f(y_0, ..., y_n, 0) = 0, f(y_0, ..., y_n, 1) = 0, f(y_0, ..., y_n, 2) = 0...$$

have solutions in naturals: if there was a solution to the tenth Hilbert problem, such an algorithm would exist, so we could also decide $K$. QED

We believe it is useful to indicate some recent developments in the research on this issue. The MRDP theorem is provable in the *Elementary Aritmetic*, namely the theory obtained from $\mathsf{I}\Delta_0$ (the theory obtained from Peano Arithmetic restricting the induction to the bounded formulas) by adding $\forall x \exists y 2^x = y$ and therefore in this theory and all its extensions, each $\Sigma_1$ formula is equivalent to a diophantine one of the form:

$$\exists x(t(x, y) = s(x, y))$$

where $t, s$ are terms of the language of *Peano Arithmetic.* By the well known essential undecidability results, for all consistent extensions of $\mathsf{Q}$, the set of $\Sigma_1$ provable sentences and the set of $\Sigma_1$sentences consistent with the theory are both undecidable. Hence also the set of Diophantine formulas *provable* in *extensions of Elementary Arithmetic* and the set of those *consistent* with these theories are undecidable too. Let us call now $D_T$ the set of Diophantine formulas consistent with the theory $\mathsf{T}$. Or, in other word the set of Diophantine equations *solvable in some models* of $\mathsf{T}$. We now ask: when $D_T$ is decidable? See Jeřábek (2016) for some positive and negative answers.

### 2.4. Creative sets and productive sets

According to Gödel, truth in the standard model cannot coincide with provability, because, as he states in a unsent letter to Yossef Balas (see Dawson (1997), p. 61):

> It follows from the correct solution of the semantic paradoxes that 'truth' of the propositions of a language cannot be expressed in the same language, while provability (being an arithmetical relation) can.

However, he adds:

> in consequence of the philosophical prejudices of our time [...] a concept of mathematical truth as opposed to demonstrability was viewed with greatest suspicion and widely rejected as meaningless.

Truth cannot be expressed because otherwise the liar paradox would be reproducible: a version of Tarski's theorem before Tarski (see Krajewski (2004) for a general analysis on the relationship between Gödel and Tarski). Once we know that provability is definable and truth is not, then assuming that provable sentences are true, i.e. that $Pr \subseteq Tr$, we get $Pr \subset Tr$ and we conclude that there must be some undecidable sentence, only we have no concrete example (see Grattan-Guinness (1979)). So, why Gödel didn't publish the indirect proof of incompleteness? The explanation is seen in the fact that Gödel was unusually cautious. He feared that relying on the concept of truth would compromise the possibility of acceptance of his results from the scientific world, because it would cause suspicion on the part of Hilbert, and of scientific and philosophical context dominated by finitists, logical positivists, formalists. In a letter to Hao Wang (see Wang (1974), p. 6), Gödel expresses his doubts:

> formalists considered formal demonstrability to be an analysis of the concept of mathematical truth and, therefore, were of course not in a position to distinguish the two.

Mathematical truth is just provability: this was the opinion of most matematicians in that time. Due to his cautious attitude, in his famous paper of 1931 Gödel mentions "truth" only in the introductory section, and the proof itself is made without the notion of truth. Truth was suspicious, provability wasn't.

The aim of this section is to account for Gödel's original intuition with the means of modern computability theory, showing the different complexity of the theorems of a theory of formal arithmetic with certain properties, and of the set of true propositions of the language of this theory. For this purpose, we refine the notion of computably enumerable set through the concept of *creative set*, introduced by Emil Post:

> The conclusion is unescapable that even for such a fixed, well defined body of mathematical propositions, mathematical thinking is, and must remain, essentially creative (Post (1944), 295).

Hence this terminology emphasises a consequence of Gödel's incompleteness theorems, namely the proof of an inherent creativity of the human mind.

*Creative sets* are those computably enumerable sets whose complement fails to be computably enumerable in a rather strong way. In some sense are those which can be shown effectively to be incomputable. Dekker's notion of *productivness*, that we will use in this version of incompleteness theorem, was based on the eralier notion of *creativeness*. Post gave versions of Gödel's theorem based on his concept of creative set. In some way Post, denying the mechanizability of human reason, anticipated the argument that human mind cannot be reduced to any set of computational rules:

> The logical process is essentially creative. This conclusion ... makes of the mathematician much more than a kind of clever being who can do quickly what a machine could do ultimately. We see that a machine would never give a complete logic; for once the machine is made we could prove a theorem it does not prove (Post (1941), 55)

To characterize the notion of *creative set*, it will be necessary to further explore the concept of *reducibility*.

**Definition 9.** (Many-one ridicibility) *Let $X, Y \subseteq \mathbb{N}$:*

1. $X \leq_m Y$, *i.e. $X$ is "many-one" reducible to $Y$, means that $x \in X$ iff $f(x) \in Y$, for some $f$ total computable. In case $f$ is injective we speak of $1-$reducibility (we write $X \leq_1 Y$).*
2. *We say that $X$ is m-complete, if it is computably enumerable and for all $Y$ computably enumerable , $Y \leq_m X$.*
3. $X =_m Y$ *iff* $X \leq_m Y$ *and* $Y \leq_m X$.

Observe that if $X \leq_m Y$ and $Y$ is computable, then also $X$ is computable: indeed, suppose that $X \leq_m Y$ through $f$; let therefore $\chi_X(z) = \chi_Y(f(z))$, where $\chi_A$ is the characteristic function of $A$.

**Theorem 20.** (Post 1944) *$X$ is computably enumerable iff $X \leq_m K$, where $K$ is the halting set.*

*Proof.* $\Rightarrow$ Let $X$ computably enumerable and let:

$$\psi(x,y) = \begin{cases} 1 & \text{if } x \in X \\ \uparrow & \text{otherwise} \end{cases}$$

By parametrization , there exists $f$ such that $\phi_{f(x)}(y) = \psi(x,y)$. Thus

$$W_{f(x)} = \begin{cases} \mathbb{N} & \text{if } x \in X \\ \emptyset & \text{otherwise} \end{cases}$$

Note that:

1. if $x \in X$, then $W_{f(x)} = \mathbb{N}$, whereby $f(x) \in W_{f(x)}$ and therefore $f(x) \in K$
2. if $f(x) \in K$ then $f(x) \in W_{f(x)}$, and thus $W_{f(x)} \neq \emptyset$ and finally $x \in X$

Hence $X \leq_m K$.

$\Leftarrow$ If $X \leq_m K$, then $X$ is computably enumerable; if $X \leq_m Y$ and $Y$ is computably enumerable, then also $X$ is. Let indeed $g : X \leq_m Y$ and let $Y$ computably enumerable; then $Y = Dom(\phi)$. Hence let $h(x) = \phi(g(x))$ whereby $x \in X$ iff $g(x) \in Dom(\phi)$ iff $\phi(g(x)) \downarrow$ iff $x \in Dom(h)$, namely:

$$X = Dom(\phi(g(x)) = Dom(h(x))$$

Hence $X$ is the domain of a partial computable function. QED

**Definition 10.** *$X$ is* productive*, if there exists a total computable function $f$ such that:*

$$W_e \subseteq X \Rightarrow (f(e) \in X \setminus W_e)$$

*The $f$ is called* productive function*. Note that if $X$ is productive , then is not computably enumerable (otherwise $X = W_e$ and we would have a contradiction).*

*We say that $X$ is* creative *, if it is computably enumerable and its complement $\overline{X}$ is productive . For instance, the halting set $K$ is creative , because is computably enumerable and its complement is productive :*

$$W_x \subseteq \overline{K} \Rightarrow id(x) \in \overline{K} \setminus W_x$$

*where $id(x) = x$. Suppose that this does not hold and that $id(x) = x \in W_x$, namely $\phi_x(x) \downarrow$ and therefore $x \in K$ (contradiction, since $W_x \cap K = \emptyset$)*

Hence a *productive set* can be "effectively" distinguished from any given computably-enumerable set. Gödel's incompleteness theorem implies that every attempt to effectively enumerate the truths of arithmetic is bound to fail: in any attempt to enumerate truth, either some falsehood is included or some truth is missed and in this case the construction permits to effectively produce the missing truth. The above definition expresses this phenomenon as a general, recursion theoretic property of sets[2].

**Theorem 21.** *If $X$ is productive e $X \leq_m Y$, then $Y$ is productive .*

*Proof.* Let $\psi$ the productive function of $X$ and let $f : X \leq_m Y$. It holds that $W_{g(x)} = f^{-1}[W_x] = \{z \in X | f(z) \in W_x\}$, $W_x \subseteq Y$, namely $f^{-1}[W_x]$ is uniformly computably enumerable in $x$. Hence $f(\psi(g(x)))$ is the productive function for $Y$; indeed:

$$W_e \subseteq Y \Rightarrow W_{g(e)} \subseteq X$$

$$\Rightarrow \psi(g(e)) \downarrow \wedge \psi(g(e)) \in X \smallsetminus W_{g(e)}$$

But $\psi(g(e)) \notin W_{g(e)}$ implies $\psi(g(e)) \notin \{z \in X | f(z) \in W_e\}$, i.e. $f(\psi(g(e)) \notin W_e$ and then:

$$f(\psi(g(e)) \downarrow \wedge f(\psi(g(e)) \in Y \smallsetminus W_e$$

QED

**Theorem 22.** *Every productive set contains an infinite computably enumerable set.*

*Proof.* Let $X$ be productive and let  be the productive function. Enumerate an infinite subset as follows:

1. let $e_0$ such that $W_{e_0} = \emptyset$; since $W_{e_0} \subseteq X$, we will have $f(e_0) \in X \smallsetminus W_{e_0}$, namely $f(e_0) \in X$. Put $y_0 = f(e_0)$.

2. Suppose we have defined $\{y_0, ..., y_n\} \subseteq X$; let $e_{n+1}$ such that $\{y_0, ..., y_n\} = W_{e_{n+1}} \subseteq X$; put $y_{n+1} = f(e_{n+1}) \in X \smallsetminus W_{e_{n+1}}$ and therefore $y_{n+1} \neq y_0, ..., y_n$.

It is observed that there is an $h$ such that $W_x \cup \{f(x)\} = W_{h(x)}$. For consider the function:

$$f_{h(x)}(y) \simeq \theta(x, y) = \begin{cases} 1 & \text{if either } y \in W_x \text{ or } y = f(x) \\ \uparrow & \text{otherwise} \end{cases}$$

Let $W_{e_{n+1}} = W_{e_n} \cup \{y_n\} = W_{e_n} \cup \{f(e_n)\} = W_{h(e_n)}$. The sequence of indices $e_n$ is therefore given computablely: $e_{n+1} = h(e_n)$ and therefore the sequence $y_n = f(e_n)$. It follows that $B = \{y_0, y_1, y_3 ...\} = \{e_0, f(h(e_0)), f(h(h(e_0))), ...\}$ is the range of a total computable function. QED

An infinite set is called *immune* if it does not contain infinite recursively enumerable sets. A set is *simple* if it is recursively (or computably) enumerable and its complement has this property. Post proved the existence of such sets. Simple sets are in intermediate between recursive and creative sets. Indeed, since a productive set contains an infinite recursively enumerable subset, a simple set is neither recursive nor creative. Now we show the equivalence between the two notions of *creativeness* and *m-completeness*.

**Theorem 23.** *If $X$ is $m$-complete, then is creative .*

2 In some textbooks the productive function is given as a total function, while in others it is given as *partial*. We follow in this exposition Odifreddi (1989-1999).

*Proof.* Let $g : K \leq_m X$ and let $W_{h(x)} = \{z | g(z) \in W_x\}$, with $X$ computably enumerable. Note that if $g(z) \in W_x \subseteq \overline{X}$, then $z \in \overline{K}$. Hence, if $W_x \subseteq \overline{X}$, then $W_{h(x)} \subseteq \overline{K}$. In particular $h(x) \notin W_{h(x)}$ and then $h(x) \in \overline{K}$. In conclusion:

$$W_x \subseteq \overline{X} \Rightarrow h(x) \in \overline{K} \setminus W_{h(x)}$$

$$\Rightarrow g(h(x)) \in \overline{X} \setminus W_x$$

Note that $g(h(x)) \in W_x \Rightarrow h(x) \in W_{h(x)}$. QED

**Theorem 24.** (Myhill 1955) *If $X$ is creative, then $X$ is $m-$complete.*

*Proof.* Let $X$ creative ; then $\overline{X}$ is productive . Let $h$ the productive function and let $Y$ computably enumerable; let us define, for $m, n$ fixed, the following algorithm: given $r$ as input, look for this $n$ in $Y$:

1. if $n$ appears, compute $h(m)$,
2. if $h(m) = r$, output $= 0$.

Otherwise the algorithm does not give output. Because it depends uniformly from $m$ and $n$, we can express it as $\phi_{f(m,n)}$ for some $f$ total computable. Hence we have:

$$W_{f(m,n)} = \begin{cases} \{h(m)\} & \text{if } n \in Y \\ \emptyset & \text{otherwise} \end{cases}$$

Now we use this version of the second recursion theorem: for every $f$ there exist a function $\nu_f$ such that $\phi_{\nu_f(x)} \simeq \phi_{f(\nu_f(x),x)}$. Let therefore $\nu_f$ be such that $W_{\nu_f(x)} = W_{f(\nu_f(x),x)}$, and then, returning to our case:

$$W_{\nu_f(n)} = W_{f(\nu_f(n),n)} = \begin{cases} \{h(\nu_f(n))\} & \text{if } n \in Y \\ \emptyset & \text{otherwise} \end{cases}$$

Hence:

1. $n \in Y \Rightarrow h\nu_f(n) \in W_{\nu_f(n)} \Rightarrow W_{\nu_f(n)} \nsubseteq \overline{X} \Rightarrow h\nu_f(n) \in X$
2. $n \notin Y \Rightarrow W_{\nu_f(n)} = \emptyset \Rightarrow h\nu_f(n) \in \overline{X}$

Hence $n \in Y$ iff $h\nu_f(n) \in X$, and therefore $h\nu_f : Y \leq_m X$. QED

Let us now return to logic, with the aim of arriving at the proof of versions of the incompleteness theorem that emphasise Gödel's original insights we mentioned at the beginning of this section on the different complexity of the set of true statements and the set of theorems, but saying something more precise about the complexity of these sets. We consider the classes of $\Delta_0$ and $\Sigma_1$ formulas of the language $\mathcal{L} = \{+, \cdot, S, 0, \leq\}$ defined at p. 13. The following properties hold already in Robinson's theory $\mathsf{Q}$:

1. If $\phi$ is un $\Delta_0$-sentence, then:
   (a) if $\mathbb{N} \models \phi$ then $\mathsf{Q} \vdash \phi$
   (b) If $\mathbb{N} \models \neg\phi$ then $\mathsf{Q} \vdash \neg\phi$
2. If $\phi$ is a $\Sigma_1$-sentence, then:
   (a) if $\mathbb{N} \models \phi$ then $\mathsf{Q} \vdash \phi$

(b) Here the "negative part" is not valid. Indeed, Gödel's undecidable sentence is just a $\Pi_1$-sentence of the form $\neg\exists x\phi(x)$.

In its constructive (syntactic) version of the first incompleteness theorem, Gödel introduces the concept of $\omega$-consistency, a property stronger than consistency, and proves that if $\mathsf{T}$ is an extesion of Robinson's arithmetic, then there exists a sentence $\phi$ of its language such that, if $\mathsf{T}$ is consistent, then $\phi$ is not provable in $\mathsf{T}$; if it is $\omega$-consistent, then also $\neg\phi$ is unprovable. The $\omega$-consistency is the property that if $\mathsf{T} \vdash \phi(\overline{n})$ for all $n$, then $\mathsf{T} \nvdash \exists x\neg\phi(x)$. However we will see that in Rosser's version the $\omega$-consistency can be dispensed, and replaced by mere consistency. We will see later with all the details that all computable relations have their "counterpart" formalized in the language $\mathcal{L}$, such that if the relation holds, then that counterpart is true. In fact, we can say more: if the relation holds, its counterpart is provable in $\mathsf{Q}$ and if the relation does not hold, then its counterpart is refutable. More formally, we will see that in all extension of Robinson's $\mathsf{Q}$, all recursive (or computable) relations are *representable* (or *binumerable*) and all recursively enumerable (or computably enumerable) relations are *weakly representable* (or *numerable*). Those representing formulas are in $\Sigma_1$. Let therefore $\tau(x, y, z)$ be the "counterpart" of Kleene's $T(x, y, z)$ in the language $\mathcal{L}$ and let $\mathsf{T}$ be an extension $\Sigma_1$-sound of $\mathsf{Q}$; then the following holds:

1. $n \in \overline{K}$ iff for all $m \in \mathbb{N}$, $\mathsf{T}(n, n, m)$ is false, iff $\mathbb{N} \not\models \tau(\overline{n}, \overline{n}, \overline{m})$, for all $m \in \mathbb{N}$, iff $\ulcorner\neg\exists y\tau(\overline{n}, \overline{n}, y)\urcorner \in Th(\mathbb{N})$.

2. $n \in \overline{K}$ iff for all $m \in \mathbb{N}$, we have that $T(n, n, m)$ is false and therefore by representability $\mathsf{T} \vdash \neg\tau(\overline{n}, \overline{n}, \overline{k})$ for all $k$ and by $\Sigma_1-$soundness (or equivalently, the 1-consistency, see at p. 108) $\mathsf{T} \nvdash \exists y\tau(\overline{n}, \overline{n}, y)$, from which $\ulcorner\exists y\tau(\overline{n}, \overline{n}, y)\urcorner \in \overline{Thm_{\mathsf{T}}}$ (the set-theoretical complement of the set of theorems).

**Theorem 25.** *Let* $\mathsf{T}$ *an axiomatizable extension and* $\Sigma_1-$*sound of* $\mathsf{Q}$*; then:*

*1. the set* $Thm_{\mathsf{T}}$ *is creative ;*

*2. the set* $Th(\mathbb{N})$ *is productive (hence not axiomatizable).*

*Proof.* Let $f(n) = \ulcorner\exists y\tau(\overline{n}, \overline{n}, y)\urcorner$ and $g(n) = \ulcorner\neg\exists y\tau(\overline{n}, \overline{n}, y)\urcorner$. What we have before verified is that:

1. $n \in \overline{K}$ iff $f(n) \in \overline{Thm_{\mathsf{T}}}$;

2. $n \in \overline{K}$ iff $g(n) \in Th(\mathbb{N})$

Recall that $\overline{K}$ is productive, and that is true in general that if a set is productive and is $m$-riducible to another set, also this will be productive. But $f : \overline{K} \leq_m \overline{Thm_{\mathsf{T}}}$ and $g : \overline{K} \leq_m Th(\mathbb{N})$; therefore $Th(\mathbb{N})$ will be productive. In addition, if $\mathsf{T}$ is axiomatizable then $Thm_{\mathsf{T}}$ is computably enumerable, and if its complement is productive, it will be creative. QED

So the two sets: the set of true statements and the set of theorems cannot coincide since they are of different complexities. Using these tools of computability theory again, we can also give a theorem of essential incompleteness and essential undecidability in Rosser's style (see Smullyan (1961), pp. 47-55). Now we work on pairs of sets:

1. $X, Y \subseteq \mathbb{N}$ are called *computablly inseparable* if there is no computable set $U$ such that $X \subseteq U$ and $U \cap Y = \emptyset$, namely, that separates them.

2. $X, Y \subseteq \mathbb{N}$ are called *effectively* inseparable, if there exists a partial computable function $\psi$ such that, if $X \subseteq W_u$, $Y \subseteq W_y$ and $W_u \cap W_v = \emptyset$, then $\psi(u, v) \downarrow\in \overline{W_u \cup W_v)}$

It holds that $X, Y \subseteq \mathbb{N}$ are effectively inseparable, then they are computably inseparable. Also it is true that $X, Y \subseteq \mathbb{N}$ are effectively inseparable, then they are creative. Such sets exists: for instance, the following sets $A = \{x|\phi_x(x) \simeq 0\}$ and $B = \{x|\phi_x(x) \simeq 1\}$ are effectively inseparable. We define indeed a $\chi(u, v, z)$ which lists simultaneously $W_u$ and $W_v$ and gives output 1, if $z$ appears first in $W_u$, while giving output 0, if $z$ appears first in $W_v$. This done, we place $\chi(u, v, z) \simeq \phi_{h(u,v)}z$. Suppose that $A \subseteq W_u$ and $B \subseteq W_v$, where $W_u \cap W_v = \emptyset$. Then

$$
\begin{aligned}
h(u, v) \in W_u &\Rightarrow \chi(u, v, h(u, v)) \simeq 1 \\
&\Rightarrow \phi_{h(u,v)}h(u, v) \simeq 1 \\
&\Rightarrow h(u, v) \in B
\end{aligned}
$$

Absurd, since $B \cap W_u = \emptyset$. Analogous, if $h(u, v) \in W_v$.

**Theorem 26.** (Rosser's essential undecidability) *Each consistent and axiomatizable extension* $\mathsf{T}$ *of Robinson's arithmetic has an undecidable sentence.*

*Proof.* Let $A, B$ effectively inseparable computably enumerable. Hence $A = W_a$, $B = W_b$; recall a theorem, due to Gödel (which will be discussed in greater detail later on), saying that computable relations are *representable* (or *binumerable*) in $\mathsf{T}$, namely, in our specific case, that we can find a formula $\tau(x, y, z)$ that represents Kleene's predicate $T(x, y, z)$, i.e. such that:

1. if $T(a, n, m)$ holds then $\mathsf{T} \vdash \tau(\overline{a}, \overline{n}, \overline{m})$
2. if $T(a, n, m)$ does not hold, then $\mathsf{T} \vdash \neg\tau(\overline{a}, \overline{n}, \overline{m})$

Now let $(A \prec B)(\overline{n})$ be the formula:

$$\exists z(\tau(\overline{a}, \overline{n}, z) \wedge \forall y \leq z \neg\tau(\overline{b}, \overline{n}, y))$$

The following applies:

1. if $n \in A$, then for some $m$, $T(a, n, m)$ is true and therefore it follows that $\mathsf{T} \vdash \tau(\overline{a}, \overline{n}, \overline{m})$ by binumerability. But $A$ and $B$ are disjoint, hence $n \notin B$ and therefore the relation $T(b, n, r)$ is false for all $r \leq m$ and for all $s \leq m$, $\mathsf{T} \vdash x = \overline{s} \to \neg\tau(\overline{b}, \overline{n}, \overline{s})$ and hence:

   $$\mathsf{T} \vdash \bigvee_{s \leq m} x = \overline{s} \to \neg\tau(\overline{b}, \overline{n}, \overline{s})$$

   Now let us consider that $Q \vdash x \leq \overline{m} \leftrightarrow x = \overline{0} \vee ... \vee x = \overline{m}$. It follows $\mathsf{T} \vdash x \leq \overline{m} \to \neg\tau(\overline{b}, \overline{n}, x)$. Hence we have proved in $\mathsf{T}$:

   $$\exists y(\tau(\overline{a}, \overline{n}, y) \wedge \forall z \leq y \neg\tau(\overline{b}, \overline{n}, z))$$

2. Analogously, if $n \in B$, then there is an $r$ such that $T(b, n, r)$ and therefore $\mathsf{T} \vdash \tau(\overline{b}, \overline{n}, \overline{r})$. Since $A$ and $B$ are disjoint we also have that $T(a, n, m)$ is false for all $m$, from which follows $\mathsf{T} \vdash \neg\tau(\overline{a}, \overline{n}, \overline{m})$. But in $\mathsf{Q}$ is provable $y \leq \overline{r} \vee \overline{r} < y$. If $y \leq \overline{r}$ then $\mathsf{T} \vdash \neg\tau(\overline{a}, \overline{n}, y)$; if $\overline{r} < y$, then $\mathsf{T} \vdash \exists z < y\tau(\overline{b}, \overline{n}, z)$. We have proved $\mathsf{T} \vdash \neg(A \prec B)(\overline{n})$, namely:

   $$\forall y(\neg\tau(\overline{a}, \overline{n}, y) \vee \exists z < y\tau(\overline{b}, \overline{n}, z))$$

Let $\mathring{A} = \{n|\mathsf{T} \vdash (A \prec B)(\overline{n})\}$ and $\mathring{B} = \{n|\mathsf{T} \vdash \neg(A \prec B)(\overline{n})\}$.

Note that $\mathring{A}$ and $\mathring{B}$ are *computably enumerable* sets: actually, for any $\theta$ and axiomatizable theory $\mathsf{T}$, the relation: "there is a proof in $\mathsf{T}$ of $\theta$", can be expressed by a $\Sigma_1-$formula. Moreover, if $\mathsf{T}$ is consistent, then these are disjoint sets. Let therefore $\mathring{A} = W_{\dot{a}}$ e $\mathring{B} = W_{\dot{b}}$. Since $A, B$ are effectively inseparable we will have a function $h(\dot{a}, \dot{b}) \in \overline{W_{\dot{a}} \cup W_{\dot{b}}}$, i.e. $h(\dot{a}, \dot{b}) \notin \mathring{A}$ and $h(\dot{a}, \dot{b}) \notin \mathring{B}$ and therefore $T \nvdash (A \prec B)(\overline{h(\dot{a}, \dot{b})})$ and $T \nvdash \neg(A \prec B)(\overline{h(\dot{a}, \dot{b})})$.

QED

Using the terminology of Smullyan (1961) we can say that such a se sentence $(A \prec B)(\overline{h(\dot{a},\dot{b})})$ *separates* $A$ *and* $B$ *in* $\mathsf{T}$. Smullyan calls "Rosser theory" a theory in which every pair of computably enumerable sets is separable in it and all axiomatizable theories that consistently extend Robinson Arithmetic actually meet this requirement. Above, in fact we have shown that this property implies essential undecidability. This further property applies to these theories $\mathsf{T}$.

**Corollary 4.** *The sets* $Thm_{\mathsf{T}}$ *and* $Ref_{\mathsf{T}}$ *of (Gödel numbers of) provable sentences and of refutable sentences of* $\mathsf{T}$ *are effectively inseparable.*

*Proof.* Recall that if $(A, B)$ and $(X, Y)$ are disjoint pairs of computably enumerable sets and $(A, B)$ is effectively inseparable and either $(A, B) \subseteq (X, Y)$ or $(A, B) \leq_m (X, Y)$, then also $(X, Y)$ is effectively inseparable, where $(A, B) \leq_m (X, Y)$ means that for some computable $f$, $f[A] \subseteq X$ and $f[B] \subseteq Y$. Indeed, let $h(x, y)$ the "productive" function for $(A, B)$. Recall that in general there is a $g$ such that $W_{g(x)} = f^{-1}[W_x]$ and define:

$$h^*(x, y) = f(h(g(x), g(y)))$$

Let therefore $X \subseteq W_i$ and $Y \subseteq W_j$ for disjoint computably enumerable sets $W_i, W_j$. Hence $A \subseteq f^{-1}[W_i]$ and $B \subseteq f^{-1}[W_j]$ where $f^{-1}[W_j] = W_{g(j)}$, $f^{-1}[W_i] = W_{g(j)}$ are disjoint. Thus, $h(g(i), g(j)) \notin W_{g(i)} \cup W_{g(j)}$, from which $h^*(i, j) \notin W_i \cup W_j$.

Now, we have that the above effectively inseparable $A, B$ satisfy $A \subseteq \mathring{A}$ and $B \subseteq \mathring{B}$. Moreover $(\mathring{A}, \mathring{B}) \leq_m (Thm_{\mathsf{T}}, Ref_{\mathsf{T}})$ via the function $f(n) = \ulcorner (A \prec B(\overline{n})) \urcorner$. QED

Some further refinement is possible. For instance, Bernardi (1981) shows that if $\mathsf{T} \nvdash \phi \leftrightarrow \psi$, then the equivalence classes $[\phi] = \{\sigma | \mathsf{T} \vdash \phi \leftrightarrow \sigma\}$ and $[\psi] = \{\sigma | \mathsf{T} \vdash \psi \leftrightarrow \sigma\}$ are effectively inseparable. Indeed, if $[\phi] \subseteq W_i$ and $[\psi] \subseteq W_j$ and $W_i \cap W_j = \emptyset$, take a formula $\alpha(x)$ which separates $W_i$ and $W_j$, i.e. such that $\mathsf{T} \vdash \alpha(\overline{n})$ if $n \in W_i$ and $\mathsf{T} \vdash \neg\alpha(\overline{n})$ if $n \in W_j$. The take the fixed point:

$$\tau \leftrightarrow ((\alpha(\overline{\ulcorner \tau \urcorner}) \wedge \psi) \vee (\neg\alpha(\overline{\ulcorner \tau \urcorner}) \wedge \phi))$$

Now observe that if $\ulcorner \tau \urcorner \in W_i$, then $\mathsf{T} \vdash \alpha(\overline{\ulcorner \tau \urcorner})$ and by the logic this implies $\mathsf{T} \vdash \tau \leftrightarrow \psi$. However $\ulcorner \psi \urcorner \in W_j$, a contradiction. A specular argument applies if $\ulcorner \tau \urcorner \in W_j$.

From the effective inseparability of the sets of theorems and of refutable formulas of $\mathsf{T}$ clearly follows the essential undecidability of $\mathsf{T}$. Indeed, suppose by contradiction that a theory $\mathsf{T}$ (consistent and r.e.) is effectively inseparable, but not essentially undecidable. Hence let $\mathsf{S}$ a axiomatizable consistent extension of it and suppose that $\psi$ is the "productive function" for the pair $(Thm_{\mathsf{T}}, Ref_{\mathsf{T}})$. Let $W_i = Thm_{\mathsf{S}}$ and $W_j$ its complement. Hence $Thm_{\mathsf{T}} \subseteq W_i$ and $Ref_{\mathsf{T}} \subseteq W_j$. It follows that $\psi(i, j) \notin W_i \cup W_j = \mathbb{N}$, which is a contradiction (for an in-depth analysis of these topics, see, for example, Cheng (2023)). The connection between the effective inseparability of theorems and inseparable statements and the incompleteness theorem can also be highlighted by resorting to the notion of effective extensibility, equivalent equivalent to effective inseparability, introduced in Boykan Pour-El (1968). Let us consider the axioms of a computably enumerable theory as a set $W_e$ of index $e$. A theory $\mathsf{T}$ is said to be *effectively extensible* if there exists a computable function $f$ such that, if $e$ is an index of a computably enumerable extension $\mathsf{S}$ of $\mathsf{T}$, then $f(e) \notin Thm_{\mathsf{S}} \cup Ref_{\mathsf{S}}$. The connection between the *effective inseparability* of theorems and refutable sentence, on the one hand, and the incompleteness theorem, on the other, can also be highlighted by resorting to the equivalent notion of *effective extensibility*, introduced in Boykan Pour-El (1968). Let us consider the axioms of a computably enumerable theory as a set $W_e$ of index $e$. A theory $\mathsf{T}$ is said to be effectively extensible if there exists a computable function $f$ such that, if $e$ is an index of a computably enumerable extension $\mathsf{S}$ of $\mathsf{T}$, then $f(e) \notin Thm_{\mathsf{S}} \cup Ref_{\mathsf{S}}$.

The issue of the different complexity between the set of true statements and the set of provable statements in certain theories can be further investigated by considering a more general notion of reducibility. Many modern computer processes are *online* interactive processes, in the sense that they interact with the environment or consult external databases (e.g. the web). In 1939 Turing sketched a formalization of this idea, through the description of an *oracle machine* ("o-machine") and in so doing invented relativisation and, in essence, the Turing jump (see Cooper (2004) and Soare (2009) for an historical overview). This idea was later considerably developed in Post (1944) and Post (1948). An oracle is a kind of black box (not necessarily a program) able to solve some problems and to produce an answer to question like: "is the element $a$ in $X$?" These machines represent an attempt to extend the power of ordinary machines and to overcome the incomputable, since they can compute something that ordinary Turing machines cannot compute. The oracle machine is a relativised model very useful in order to compare and classify degrees of undecidability of problems, and to define the *Arithmetical Hierarchy*.

We introduce a more general concept of reducibility, based on a model of machine that allows the consultation of an *oracle* during the course of computation. A limitation of the notion of $m$-reducibility that requires a correction can be derived from this example. If $B \leq_m A$, it seems natural to ask that $A$ encompasses the information contained in $B$. In some sense $A$ contains also the information about $\overline{A}$, yet there is no computably enumerable *not recursive* set such that $\overline{A} \leq_m A$. Suppose, on the contrary, that such a set exists, i.e. that there exist an $f$ such that $n \in \overline{A}$ iff $f(n) \in A$. If $A$ is computably enumerable, then $A = W_e$; hence $\phi_e(f(n)) \downarrow$ iff $f(n) \in A$ iff $n \in \overline{A}$; therefore $\overline{A} = Dom(\phi_e(f(x))$ and $\overline{A}$ would be computably enumerable and therefore $A$ would be recursive, against the hypothesis.

To introduce this new model, let us consider for example our first very basic model of a Turing machine, where we made these conventions.

1. Input convention: To input $n$, place $n + 1$ consecutive 1's on the tape.
2. Output convention: If a computation halts — which only happens when there is no applicable quadruple in the program — output the number $f(n)$ of 1's left printed on the tape.

We can think of a machine with oracle for $A$ as equipped, in addition of the input tape, of a further read-only tape, which contains the characteristic function $\chi_A(x)$ of $A$ – substantially a binariy string, and the oracle tape head begins on the cell containing $\chi_A(0)$.

The oracle machine is based on instructions of the type $q_i \alpha q_j q_e$ ("You are in the state $q_i$ reading $\alpha$, count the number $n$ of 1 on the input tape and ask the oracle if $n \in A$; if the answer is yes, go in the state $q_j$, otherwise, go in the state $q_e$").

**Definition 11.** *Let $B, A \subseteq \mathbb{N}$; we say that $B$ is computable from $A$ if we can answer the question "$n \in B$?" by means of an algorithm that may have available a finite number of answers to questions about the membership of $A$, i.e. to questions of this form:*

$$n_0 \in A?, ..., n_k \in A?$$

*We write $B \leq_T A$ ("$A$ computes $B$").*

Note that if $B \leq_m A$, then $B \leq_T A$. Since any description of an oracle Turing machines is finite, it is possible to effectively encode them with natural numbers. We write, using uppercase (to emphasise that they are functional, rather than functions) Greek letters, $\Phi^A(x)$ to denote the computation of the $e - th$ oracle Turing machine with oracle $A$ on input $x$. The $m$-reducibility can be seen as a Turing-reducibility in which we are allowed to ask $A$ just once "$f(n) \in A$?". Note that now $\overline{A} \leq_T A$: indeed, let $\Phi^A(n)$ the function that outputs 1 if $n \in A$, and 0 if $n \notin A$, noting that $\chi_{\overline{A}} \simeq \Phi^A$.

**Definition 12.** *We say that:*

1. *$\psi$ is $A - recursive$, iff there exist an $e$ such that $\psi \simeq \Phi_e^A$*
2. *$X$ is recursive in $Y$, i.e. $X \leq_T Y$, iff the characteristic function $\chi_X$ is $Y - recursive$, i.e. there is an $e$ such that $\chi_X(y) \simeq \Phi_e^Y(y)$. For simplicity we will write $X(y) \simeq \Phi^Y(y)$, identifying $X$ with its characteristic function.*
3. *$X$ is computably enumerable in $Y$, if $X$ is empty, or $X = Cod(f)$ and $f \simeq \Phi_e^Y$, for some $e$ (equivalently, $X = W_e^Y$, for some $e$).*

4. *The degree of a set $A$ is the equivalence class $deg(A) = \{X | X =_T A\}$.*

We can now formulate a *relativised* version of the Church-Turing thesis which has this shape:

> *Post-Turing thesis*: $B$ is effectively *computable from* $A$ iff $B$ is computable by a Turing machine *with oracle in* $A$.

Degrees are ordered by the relation $deg(A) \preccurlyeq deg(B)$ iff $A \leq_T B$. Let $< \mathbb{D}, \preccurlyeq >$ the set of degrees with this relation. In particular, there is a minimum degree (that of recursive sets $deg(\emptyset)$, denoted by $\mathsf{0}$, but there is no maximum degree, $< \mathbb{D}, \preccurlyeq >$ is not linearly ordered and it is an upper semilattice: that is to say, given two elements, exists the supremum, but there are pairs of degrees that do not have the infimum.

**Definition 13.** *The* jump $A'$ *of* $A$ *is defined as follows:*

1. $A' = \{x | \Phi_x^A(x) \downarrow\} = K^A$.
2. $A^{n+1} = (A^n)'$
3. $A^\omega = \{\langle m, n \rangle | m \in A^n\}$

Let $\mathsf{a}^n = deg(A^n)$; the level $\mathsf{0}'$ is therefore that of $K$, of creative sets as the theorems of axiomatizable theories of formal arithmetic (e.g. *Peano arithmetic*). We will see that the set of the true statements of these theories is $\mathsf{0}^\omega$. Given two degrees $\mathsf{a} = deg(A)$ and $\mathsf{b} = deg(B)$, always exists $sup\{\mathsf{a}, \mathsf{b}\}$ and is defined as $\mathsf{a} \vee \mathsf{b} = deg(A \oplus B)$, where $A \oplus B = \{2x | x \in A\} \cup \{2x+1 | x \in B\}$. This does not apply for the infimum.

In fact, for a result of Kleene, Post and Spector, the sequence $\mathsf{0}, \mathsf{0}', \mathsf{0}'', ...$ has an "exact pair", that is, a couple of degrees $\mathsf{a}$ and $\mathsf{b}$ such that:

1. for all $n \in \mathbb{N}$, $\mathsf{0^n} \preccurlyeq a$ and $\mathsf{0^n} \preccurlyeq b$
2. for all $\mathsf{d}$, $\mathsf{d} \preccurlyeq \mathsf{a}$ and $\mathsf{d} \preccurlyeq \mathsf{b}$ implies that exists $n \in \mathbb{N}$, $\mathsf{d} \preccurlyeq \mathsf{0^n}$.

It follows that $\mathsf{a}$ and $\mathsf{b}$ can not have an infimum.

In Post (1944) the Polish-American logician posed the problem od determining if there are sets $A, B$ computably enumerable incomparable with respect to $\leq_T$, i.e. $A \not\leq_T B$ and $B \not\leq_T A$. After in 1954 Kleene and Post had proved that there exist sets $A, B$ (not necessarily computably enumerable) incomparable and such that $A, B \leq_T \emptyset'$, in the mid-1950s the following theorem was finally proved.

**Theorem 27.** (Friedberg-Muchnik 1956-1957) *There are sets $A, B$ computably enumerable incomparable (hence $\emptyset \leq_T A, B \leq_T \emptyset' = K$, and being $A, B$ incomparable, actually we have $<_T$).*

*Proof.* (see Cooper (2003), pp. 238-41 for a proof). QED

In what follows, we must investigate the relationships between *computability* and *definability* (in the standard model). For this reason, we must first dwell on the *Arithmetical Hierarchy*, more than we have already anticipated. Let us consider formulas of the language of a first order theory of formal arithmetic. We start with the following classification[3].

*Arithmetical hierarchy of formulas.* Let us return briefly to the hierarchy of formulas introduced at p. 13, pointing out only that some authors use a different, *extended*, definition of the arithmetical hierarchy, in which bounded quantifiers are considered immaterial when counting the complexity of a formula: for example, $\Sigma_{n+1}$ is the set of formulas obtained by prepending an arbitrary block of existential quantifiers *and bounded universal* quantifiers to $\Pi_n$ -formulas.

A relation $R$ is *arithmetical* if there is a formula $\theta$ of the language of Peano (and of Robinson) arithmetic which defines it in the standard model, i.e. such that:

$$\mathbb{N} \models \theta(\overline{n_0}, ..., \overline{n_k}) \text{ if and only if } \langle n_0, ..., n_k \rangle \in R$$

[3] The correct notation would be $\Sigma_n^0$, $\Pi_n^0$, where the superscript 0 means "first order"; as we deal only with these formulas, we omit this symbol for simplicity.

where $\overline{n_j}$ is the term which denotes the number $n_j$. Accordingly, a relation $R$ is $\Sigma_n-$definable, if it is definable by a formula of complexity $\Sigma_n$ and so on. However, we would like to warn the reader against possible misunderstandings, because in the scientific literature there are different conventions with regard to the level $\Delta_0 = \Sigma_0 = \Pi_0$ in the hierarchy of sets:

1. *In computability theory*: at this level there are just the computable relations. In Odifreddi (1989-1999) pp. 363-73 this convention is adopted: extend the language with symbols for every recursive relation; the intended model is the standard model expanded with all recursive relations. Being the computable relations first-order definable in the language of arithmetic, we may suppose that, for each computable relation, the language contains a relation symbol.

2. *In formalized in arithmetic*: at the zero-level there are the relations definable by formulas in which the quantifiers that are allowed to appear are only *bounded* quantifiers. Hence, what there is actually depends on the language.

Some closure properties allow to simplify the definition. Since in the standard model $\langle \mathbb{N}, +, \cdot, S, 0, < \rangle$ holds the collection (or replacement) scheme:

$$\forall x \leq t \exists y \theta(x, y) \leftrightarrow \exists z \forall x \leq t \exists y \leq z \theta(x, y)$$

saying that a bounded quantifier can be 'pushed inside' an unbounded quantifier, the above definitions (with or without bounded quantifiers interpsersed) are equivalent *in the model.* If the theory in which we work is strong enough to prove it, the equivalence holds in *the theory* too. We can also $\Delta_0$-define the *pairing relation* $\langle x, y \rangle = z$ in the language of Peano arithmetic as $2z = (x + y + z)(x + y) + 2x$. With this, we can contract two quantifiers of the same sort into one, e.g. $\exists x \exists y \theta(x, y)$ becomes $\exists v \forall x \leq v \forall y \leq v(2v = (x + y + z)(x + y) + 2x \rightarrow \theta(x, y))$. In conclusion, in definability, the alternations of quantifiers can be be understood as alternations of *blocks* of universal or existential quantifiers and each block of quantifiers of the same species can be contracted to only one quantifier of that species. The relativised arithmetic hierarchy $\Sigma_n^A$, $\Pi_n^A$ is defined as above, except that the matrix $R$ of a formula in prenex form $Q_0 x_0 ... Q_n x_n R(x_0, ..., x_n)$ instead of being recursive, is $A-$recursive, i.e. $R \leq_T A$. More exactly:

1. $\Sigma_0^A = \Pi_0^A = \Delta_0^A =$ sets recursive in $A$.
2. $\Sigma_{n+1}^A =$sets definable by formulas of the form $\exists x R(x, y)$ with $R \in \Pi_n^A$.
3. $\Pi_{n+1}^A =$sets definable by formulas of the form $\forall x R(x, y)$ with $R \in \Sigma_n^A$.
4. $\Delta_{n+1}^A = \Sigma_{n+1}^A \cap \Pi_{n+1}^A$

Many results "relativise" by substituting the concept of *recursiveness* with its version relativised to a given set. For instance $B \in \Sigma_1^A$ if and only if $B = W_e^A$.

We now come to an important result for the purpose of this section, namely Post's theorem. Though Post did not publish this theorem with a proof, Kleene (1952) credits Post with the idea.

**Theorem 28.** (Post's Theorem) *The following relations hold:*

1. $B \in \Sigma_{n+1}$ *iff* $B$ *is computably enumerable in some* $A \in \Sigma_n$.

2. $\emptyset^{n+1}$ *is* $\Sigma_{n+1}$*-complete, namely is itself* $\Sigma_{n+1}$ *and for all* $A \in \Sigma_{n+1}$, $A \leq_m \emptyset^{n+1}$.

3. $B \in \Sigma_{n+1}$ *iff* $B$ *is computably enumerable in* $\emptyset^n$.

4. $A \in \Delta_{n+1}$ *iff* $A \leq_T \emptyset^n$.

5. *Hence, in particular:*

   (a) $A \in \Delta_1$ *iff* $A \leq_T \emptyset$.

   (b) $A \in \Delta_2$ *iff* $A \leq_T \emptyset'$

*Proof.* The proof requires some preliminary lemma, which we will see now. First let's make these simple observations:

1. If $A$ is computably enumerable in $B$ and $B \leq_T C$, then $A$ is computably enumerable in $C$. Indeed, since $B \leq_T C$ iff $B(x) \simeq \Phi_e^C(x)$, then, if $A = W_a^B$, we can consider $C$ as another oracle that we can consult to know if $n \in B$ and then $A = W_j^C$.

2. $B$ is computably enumerable in $A$ iff $B$ is $\Sigma_1^A$. This is a relativised version of a well-known result.

3. $B$ is computably enumerable in $A$ iff $B$ is computably enumerable in $\overline{A}$. It follows from the first point, and from the fact that $\overline{A} \leq_T A$.

QED

**Lemma 4.** *$B$ is computably enumerable in $A$, iff $B \leq_m A'$.*

*Proof.* $\Leftarrow$ If $f : B \leq_m A'$, then $x \in B$ iff $f(x) \in A'$ iff $\Phi^A_{f(x)}(f(x)) \downarrow \simeq \psi^A(x)$. Ergo $B = Dom(\psi^A(x))$. $\Rightarrow$ Let $B = W_e^A$; let us remember that if $B = W_e^\emptyset$ then $B \leq_m \emptyset' = K$ (completeness of di $K$). More generally, relativizing, $B = W_e^A$ implies $B \leq_m A'$. QED

**Lemma 5.** *$B \leq_T A$ iff $B, \overline{B}$ are computably enumerable in $A$.*

*Proof.* $\Leftarrow$ If $B = W_e^A$ and $\overline{B} = W_i^A$, take the algorithm $\xi(e, i, n)$ that runs simultaneously $\Phi_e^A(n)$ e $\Phi_i^A(n)$. One of the two halts and answers the question if $n \in B$, hence $B \leq_T A$. $\Rightarrow$ Exercise. QED

**Definition 14.** . *$\phi_{e,s}(x) = y$ iff $x, y, e < s$ and $y$ is the output of $\phi_e(x)$ obtained in less than $s$ steps.*

**Theorem 29.** *$B \in \Sigma_{n+1}$ iff $B$ is computably enumerable in some set $\Sigma_n$ iff $B$ is computably enumerable in some set $\Pi_n$.*

*Proof.* We follow Cooper (2003) pp. 154-57 in proposing rather a sketch of the proof to highlight the key points. $\Rightarrow$ $x \in B$ iff $\exists y A(x, y)$ is true, for some $A \in \Pi_n$ (notation: $B \in \Sigma_1^A$), namely is computably enumerable in $A$; but $A \leq_T \overline{A}$ and therefore $B$ is computably enumerable also in $\overline{A}$ and $\overline{A} \in \Sigma_n$. $\Leftarrow$ If $B = W_e^A$ for $A \in \Sigma_n$, then $x \in B$ iff:

1. "there exists an $s$.
2. there exist finite *positive* answers $y_0 \in A, ..., y_m \in A$,
3. there exist finite *negative* answers $x_0 \in \overline{A}, ..., x_k \in \overline{A}$,

that allow us to determine whether $x \in W_{e,s}^A$".

We remark that 2. is $\Sigma_n$, 3. is $\Pi_n$, e.g. for $n = 3$ the conjunction of 2. and 3. has the form $\exists y_0 ... \exists y_m [\exists\forall\exists \wedge ... \wedge \exists\forall\exists] \wedge \exists x_0 ... \exists x_k [\forall\exists\forall \wedge ... \wedge \forall\exists\forall]$ that is equivalent to $\exists\forall\exists\forall$.

Lastly, "$x \in W_{e,s}^A$" is a computable relation and the entire expression in quotes is $\Sigma_{n+1}$ QED

**Theorem 30.** *For all $n > 0$, $\emptyset^n$ is $\Sigma_n-$complete.*

*Proof.* For $n = 1$ obvious, because $\emptyset' = K$; if $n > 1$ suppose by induction hypothesis that $\emptyset^n$ is $\Sigma_n-$complete; note that $e \in \emptyset^{n+1}$ iff $\Phi_e^{\emptyset^n}(e) \downarrow$ iff $e \in W_e^{\emptyset^n}$, namely $\emptyset^{n+1}$ is computably enumerable in $\emptyset^n$, that by hypothesis is $\Sigma_n$ complete. Hence from the previous theorem it follows that $\emptyset^{n+1}$ is $\Sigma_{n+1}$. In addition, it is complete: in fact suppose that $B \in \Sigma_{n+1}$, hence is computably enumerable in $\Sigma_n$ and by the induction hypothesis $\emptyset^n$ is $\Sigma_n-$complete; it follow that $B$ is computably enumerable in $\emptyset^n$; but for the above lemmas we have that this is true iff $B \leq_m (\emptyset^n)' = \emptyset^{n+1}$. QED

**Theorem 31.** *$B \in \Sigma_{n+1}$ iff $B$ is computably enumerable in $\emptyset^n$.*

*Proof.* $\Leftarrow$ If $B$ is computably enumerable in $\emptyset^n$, then, being $\emptyset^n \in \Sigma_n$, $B$ is computably enumerable in a set $\Sigma_n$. But this means that $B \in \Sigma_{n+1}$. $\Rightarrow$ if $B \in \Sigma_{n+1}$, then $B$ is computably enumerable in some $A \in \Sigma_n$; but $\emptyset^n$ is $\Sigma_n$−complete and therefore $A \leq_T \emptyset^n$ e $B$ is computably enumerable in $\emptyset^n$. QED

**Theorem 32.** *$B \in \Delta_{n+1}$ iff $B \leq_T \emptyset^n$.*

*Proof.* $B \in \Delta_{n+1}$ iff $B \in \Sigma_{n+1}$ and $B \in \Pi_{n+1}$, namely $\overline{B}, B \in \Sigma_{n+1}$, iff $\overline{B}, B$ are computably enumerable in $\emptyset^n$, iff $B \leq_T \emptyset^n$. QED

We give another indirect version of Gödel theorem, from which the distance, in terms of complexity, between the set of true statements and the set of theorems of an axiomatisable theory becomes even more evident, considering that the set of theorems is placed at the level of the first jump of the empty set.

**Theorem 33.** $\emptyset^\omega \equiv_T Th(\mathbb{N})$.

*Proof.* 1. $\emptyset^\omega \leq_T Th(\mathbb{N})$. Remember that $A'$ is computably enumerable in $A$, in particular $\emptyset^{n+1} = W_e^{\emptyset^n}$. For E. Post theorem we have then $\emptyset^{n+1}$ is $\Sigma_{n+1}$-definable:

$$m \in \emptyset^{n+1} \Leftrightarrow \mathbb{N} \models \overbrace{\exists x_0 \forall x_1 \exists x_2 ...}^{n+1-times} R(x_0, ..., x_n, m)$$

Take therefore $h(m) = \ulcorner \overbrace{\exists x_0 \forall x_1 \exists x_2 ...}^{n+1-times} R(x_0, ..., x_n, m) \urcorner$. Clearly $\langle m, n \rangle \in \emptyset^\omega$ iff $m \in \emptyset^n$ iff $h(m) \in Th(\mathbb{N})$, i.e. $\emptyset^\omega \leq_T Th(\mathbb{N})$.

1. $\emptyset^\omega \geq_T Th(\mathbb{N})$. Consider a formula in the prenex form $Q_0 x_0, ..., Q_n x_n \theta$, where each $Q_j$ is a quantifier. We transform it in a $\Sigma_{n+1}$-formula in this way:

$$\exists y Q_0 x_0, ..., Q_n x_n (\theta \wedge y = y \wedge z = z)$$

Let now $B = \{c | \exists y Q_0 x_0, ..., Q_n x_n (\theta \wedge y = y \wedge c = c)\}$. Note that:

(a) If $Q_0 x_0, ..., Q_n x_n \theta$ is true, then $B = \mathbb{N}$

(b) If $Q_0 x_0, ..., Q_n x_n \theta$ is false, then $B = \emptyset$

We give a sketch of the proof (see Rogers (1987) p. 318 for further details). Since $B \in \Sigma_{n+1}$, then for Post's results it is computably enumerable in $\emptyset^n$, namely $B = W_e^{\emptyset^n}$. Moreover we know that if $B$ is computably enumerable in $A$, then $B \leq_T A'$: more exactly , we can uniformly find an index $e$ of $B$ as a set computably enumerable in $\emptyset^n$, and from this an index $f(e)$ such that $\phi_{f(e)} : B \leq_m A'$. Hence, if $B = W_e^{\emptyset^n}$, then $\phi_{f(e)} : B \leq_m \emptyset^{n+1}$. Lastly, $\mathbb{N} \models Q_0 x_0, ..., Q_n x_n \theta$ iff $0 \in B$ iff $\phi_{f(e)}(0) \in \emptyset^{n+1}$ iff $\langle \phi_{f(e)}(0), n+1 \rangle \in \emptyset^\omega$, namely $Th(\mathbb{N}) \leq_T \emptyset^\omega$.

QED

## 2.5. Guide for further studies: trial-and-error machines

The positions of Alan Turing and Kurt Gödel appear somewhat paradigmatic in the debate about the relationships between computability and the mind. While reiterating his unconditional admiration for the work of the English mathematician, Gödel attributed a "philosophical error" to Turing, which in his view consisted in the belief that mental procedures cannot go beyond mechanical procedures. Turing's argument rested - according to Gödel - on the assumption that a finite mind is only capable of a finite number of distinguishable states. If we admit an infinity of mental states - says Turing - some of them will be arbitrarily close and therefore confused (see Turing (1936), p. 250), whereas on the contrary, according to Gödel, "the mind, in its use, is not static, but in continuous development":

> Therefore, although at each stage of the mind's development the number of its possible states is finite, there is no reason why this number should not converge to infinity in the course of its development (Letter to Hao Wang, in Wang (1974), p. 325).

A closer examination, however, reveals how Gödel's criticism of Turing is misleading, especially if one takes into account his post-war writing on mind (see Copeland and Shagrin (2013) for a thorough discussion), where Turing began to deepen the idea of *learning*:

> What we want is a machine that can learn from experience. The possibility of letting the machine alter its own instructions provides the mechanism for this. One can imagine that after the machine had been operating for some time, the instructions would have altered out of all recognition (Turing (1947), p. 393).

He conceived a Multi-Machine theory of mind, or the transformation of one Turing machine into another. A machine with the ability to learn is able to modify its table of instructions, transforming itself into a different Turing machine. The notion of mind change was later used by Putnam in developing his notion of "trial and error predicates". Trial-and-error machines where introduced in Putnam (1965) and in Gold (1967), and based on the idea of computability in the limit, i.e. a type of computation performed by an ideal model which proceeds by changing its opinion a finite number of times about the membership of a number to a set, but stabilized to the limit (hence going so far beyond the classical boundaries of computability) with the aim to represent a cognitive phenomenon like language learning and actually these writings had a strong influence on the development of the formal study of the process of gaining information through observation (the *Formal Learning Theory*, see Osherson, Stob and Weinstein (1986) and Kelly (2023)). Some researchers in cognitive science and philosophy of mind, even go so far as to claim that humans are automata of this sort, namely trial-and-error machines (see e.g. Kugel (1986)). The trial-and-error model transcends the Church-Turing thesis; indeed a similar machine solves the *halting-problem*: let for example $U$ be a machine that, when receives as input the code $e$ of another machine $\phi_e$ and a number $n$, returns immediately 0 (to say that $\phi_e(n)\uparrow$), then it starts to compute $\phi_e(n)$. If later, at a computation step $\phi_e(n)\downarrow$, $U$ then change its minds and writes 1. Sets calculated from these kind of machines turn out to be the $\Delta_2$, and not only the $\Delta_1$, (i.e. the *recursive* ones). However, it must be emphasised that this is a purely ideal model, since we have no way to know at any given time whether the latest output is the correct output.

In the 1970s, Magari (1974) and Jeroslow (1975) proposed two further formal counterparts of the concept of "trial and error theory"; in particular, Magari's so-called *dialectical sets* form a strict subclass of the $\Delta_2$ sets. Thinking of Gödel's limitative results, Magari's purpose, in the light of Lakatos (1976) dialectical reconstruction of history of mathematics, influenced by Popper's fallibilism, was in particular that of introducing a kind of formal systems, consisting of two actions: on the one hand removing contradictions when they arise, by removing some axioms, and on the other hand adding axioms until they do not give rise to contradictions. In more recent years, this idea has been developed in a series of works (see Amidei, Pianigiani, San Mauro, Simi and Sorbi (2016) and the other articles by these authors, cited in the bibliography).

Here we just want to demonstrate a major achievement in this area, as a starting point for the study of this topic. A set $A$ will be identified by its characteristic function, that is with the infinite binary sequence $A(0), A(1), A(2), ...,$ where $A(n) = 1$ iff $n \in A$. We denote $A \upharpoonright y$ the restriction of the characteristic function of $A$ to the initial segment of elements $z < y$, $A(0), A(1), ..., A(y-1)$ To say that a set $A$ is computably enumerable is to say that it has a computable approximation (see p. 15), namely that there exists a computable function $f$ such that $A(x) = \lim_{s\to\infty} f(x, s)$, $f(n, 0) = 0$ and for at most a $s$, we have that $f(n, s) \neq f(n, s+1)$, i.e. the function, changes its mind about $n$ at most once. Note that if we put $f(n, s) = A_s(n)$ this means that that $A_s \subseteq A_{s+1}$ (monotonicity of the sequence). Now we *generalise* this definition, admitting that $f$ may change its mind a *finite number of times* (thus abandoning the *monotonicity*). Let's say that $\{A_s\}_s$ is a $\Delta_2$-approximation of $A$, if $A = lim_{s\to\infty} A_s$, that is approximated by a computable sequence, where an element can enter and exit a certain finite number of times. Call *modulus of convergence* for such a sequence, a function $m(x)$ such that, for all $x$ and all $s \geq m(x)$, $A_{\upharpoonright x+1} = A_{s\upharpoonright x+1}$. That is to say, after the stage $m(x)$ the initial segment of $A$ until $x$ does not change any more. In what follows we shall consider in particular the *minimum* modulus:

$$m(x) = \mu(s).\forall t \geq s(A_{\restriction x+1} = A_{t\restriction x+1})$$

Note that if $A = lim_{s\to\infty} A_s$, then $A \leq_T m$; actually $A(x) = A_{m(x)}(x)$. On the other hand, if we consider the computably enumerable set (called *the set of changes*):

$$B = \{\langle x, s\rangle | \exists t > s(A_s(x) \neq A_t(x))\}$$

then clearly $m \leq_T B$. In fact, also $m \leq_T B$ holds. So ultimately $m =_T B$, namely, the minimum modulus has Turing degree computably enumerable.

**Lemma 6.** (Shoenfield's Limit Lemma 1959) *$A$ is computable in the limit iff $A \leq_T \emptyset^{'}$ (iff, by Post's results, $A \in \Delta_2$).*

*Proof.* $\Leftarrow$ suppose that $A = \Phi^{\emptyset^{'}}$; let $g$ a function with values $0 - 1$ such that $g(n, s) = 1$ iff $\Phi_s^{\emptyset^{'}}(n) = 1$. Let $z = \mu x.\Phi^{\emptyset^{'}\restriction x}(n)\downarrow$ the so-called *use* of the computation (i.e. the length of the minimal initial segment of the oracle sufficient for computing the function on n). Recall that the *Use Principle* says: if $z$ is the use of $\Phi^A(n)$ and B is such that $B \restriction z = A \restriction z$, then $\Phi^A(n) = \Phi^B(n)$. But $\emptyset^{'}$ is computably enumerable and therefore can be approximated by a sequence $\emptyset^{'}_s \subseteq \emptyset^{'}_{s+1}$. Let therefore $s$ be a sufficiently large stage, for which $\emptyset^{'} \restriction z = \emptyset^{'}_s \restriction z$. For the "Use Principle", this means that $g(n, t) = \Phi^{\emptyset^{'}}(n) = \Phi^{\emptyset^{'}_t}(n)$, for all $t \geq s$. Take $A(x) = \lim_t g(x, t)$.

$\Rightarrow$ Suppose that $A(x) = \lim_t g(x, t)$, where $g(x, 0) = 0$. Notes that this means that, if I count the number $k$ of changes of opinion by $g$, the function returns value 1 only when $k$ is odd. Consider now the set $B$ of pairs $\langle n, k\rangle$ such that the numbers $s$ of stages in which $g(n, s) \neq g(n, s + 1)$ is greater or equal to $k$ and note that it is computably enumerable (therefore $B \leq_T \emptyset^{'}$). Consider a $\Phi^B$ working as follows. On input $n$, look for the minimum $k$ such that $\langle n, k\rangle \notin B$: if you find it, output 0 in case $k$ is even, and output 1 in case is odd. Note that $A(x) = \Phi^B(x)$ and therefore $A(x) = \Phi^{\emptyset^{'}}(x)$. QED