All pure mathematics follows formally from twenty premisses.

(Bertrand Russel,
The Principles of Mathematics)

Part C

Bounded Arithmetic

Chapter V

Bounded Arithmetic

In the previous chapters we have encountered some weak fragments of arithmetic. First we have seen that some properties of numbers can be proved and several concepts can be formalized even in very weak fragments. Often we have used $I\Sigma_1$ as a base theory where a weaker theory was possible. In this chapter we shall consider the possibility of formalizing syntax and some basic combinatorial concepts in weaker fragments. Section 3 will be devoted to this aim. Let us stress that the reason for trying to formalize certain concepts in weak fragments is not only our curiosity. In fact weak fragments are often needed in the study of strong theories. Recall the concept of a definable *cut*, Chap. III, Sect. 3. Cuts in fragments of *PA* which are closed under $+$ and $*$ define interpretations of bounded arithmetic ($I\Sigma_0$ and some stronger systems), but usually they do not interpret even $I\Sigma_1$. Thus we need weak fragments in order to understand the theory of cuts.

The second reason, which is perhaps the main reason for this research into weak fragments, is their close connection to computational complexity. It is obvious that there is some connection. In $I\Sigma_0$ we have induction for classes of bounded formulae. It is well-known that bounded formulae define easily-computable sets; for instance they are computable in linear space. This is however only a superficial observation. Some deeper connections will be shown in Chap. IV. It seems that more such relations are yet to be discovered.

Some knowledge of complexity belongs to a general background of all working mathematicians. Even so, we have decided to include Sect. 2 as a short introduction in complexity theory, since concepts like the Polynomial Hierarchy are quite often disregarded in textbooks.

Section 1 has a similar introductory role. We shall survey the most important weak fragments and prove some basic results on them. Note that some results on such theories (especially Q and I_{open}) have been proved in Chap. I, Sect. 1.

The last section, Sect. 5, deals with classical metamathematical questions in the context of bounded arithmetic: unprovability of consistency, partial conservativity, interpretability. We will also have to develop some proof theory in Bounded Arithmetic.

In the proofs we shall not favor a particular proof technique; we shall mix proof theory with model theory, since our aim is not the study of one of these fields, but the study of weak fragments. An exception is Sect. 4, where we prefer model theory, since the same subject has been treated using proof theory in another book, [Buss 86, Bounded Arith.]; this section however contains several results not included in Buss's book.

This chapter needs very little from the previous chapters and hence can be read almost independently. It requires only basic acquaintance with fragments of Peano arithmetic, Chap. I, Sects. 1 and 2. In Sect. 5 we shall work with definable cuts, thus we need Chap. III, Sect. 3.

1. A Survey of Weak Fragments of Arithmetic

(a) Fragments of Arithmetic

The weakest fragment that we consider in this book is ***Robinson's arithmetic*** Q. It was defined in Chap. I, Sect. 1. The fragments of arithmetic which use the language L_0 are extensions of Q by adding a schema of induction for a restricted class of formulae and, possibly, an axiom saying that a certain function is total. Though Q proves only very few interesting statements about numbers, (it does not prove for instance the commutativity of addition), we shall see that quite strong theories are ***interpretable*** in it, (Sect. 5).

Sometimes it is more convenient to start with a stronger theory. Usually one takes P^- which is just the axiomatization of the ***nonegative part of the discretely ordered ring***. These axioms are provable in I_{open} and they are listed in Theorem 1.10, Chap. I, where we only have to add

$$\overline{0} \leq x \,\&\, x \leq \overline{1} \rightarrow . \, x = \overline{0} \vee x = \overline{1} .$$

The next step in the hierarchy is I_{open}. Several basic properties of natural numbers are provable already in this theory, see Chap. I, Sect. 1. In spite of this fact it is a very weak theory. There are models of I_{open} in which the following equations do have solutions

$$(x+1)^2 = 2(y+1)^2 ;$$
$$(x+1)^3 + (y+1)^3 = (z+1)^3 ;$$

i.e. the irrationality of $\sqrt{2}$ and Fermat's last theorem for exponent 3 are not provable. Such independence results are possible, because there are recursively defined models of I_{open}. It seems that the properties such as the property of having recursive models determine the border-line between very weak fragments and other fragments of arithmetic. I_{open} is an interesting

object for research. However the methods needed for it are different from those needed for stronger systems, therefore we shall not consider it here. Let us only mention here that one can prove independence results also for some strengthenings of I_{open} (as it is defined here). Then it is more convenient to define I_{open} to be a ring, not only the nonegative part of it, and add properties such as *normality* (which means that it is integrally closed in its fraction field and which implies that $\sqrt{2}$ *is* irrational).

The next natural step in the hierarchy is $I\Sigma_0$; let us recall that this is Q with induction schema for all bounded formulae in language L_0. First we state two theorems which show that there is an essential difference between the structure of models of $I\Sigma_0$ and I_{open}, since none of these theorems holds for I_{open}.[1] The proofs require some facts which will be proved later.

1.1 Theorem. There is no recursive nonstandard model of $I\Sigma_0$; in fact, in each nonstandard countable model M of $I\Sigma_0$ both $+_M$ and $*_M$ are not recursive.

Proof. Let A, B a recursively inseparable pair of subsets of N. Let φ and ψ be Σ_0 formulae such that

$$n \in A \equiv N \models (\exists y)\varphi(\overline{n}, y),$$
$$n \in B \equiv N \models (\exists y)\psi(\overline{n}, y).$$

Since A and B are disjoint, we have

$$N \models (\forall x, y, z)\neg(\varphi(x, y) \,\&\, \psi(x, z)).$$

Hence, for every k,

$$M \models (\forall x, y, z \leq k)\neg(\varphi(x, y) \,\&\, \psi(x, z)).$$

Since this is a Σ_0 formula, by overspill we get some $a \in M$ nonstandard such that $(\forall x, y, z \leq a)\neg(\varphi(x, y) \,\&\, \psi(x, z))$ holds in M. Let C be defined by

$$n \in C \equiv M \models (\exists y \leq a)\varphi(\overline{n}, y).$$

Then $A \subseteq C$ and $C \cap B = \emptyset$. Let C' be an initial segment of C. We shall code it by an element c' of M in such a way that b belongs to C' iff the b-th prime divides c'. Clearly this coding can be expressed by a Σ_1 formula. Using a similar overspill argument as above one can show that there is a $c \in M$ which codes the whole C, i.e.

$$n \in C \equiv M \models \overline{p_n} \mid c,$$

[1] Let us note that R. Kaye has defined a fragment of arithmetic for which 1.1 holds, but for which 1.2 fails.

where p_n is the n-th prime. Now assuming that $+_M$ is recursive, we can show that C is recursive: given n, search for $r, s \in M$, $r <_M s$, such that

$$M \models c = \underbrace{s + s + \cdots + s}_{p_n \text{ times}} + r .$$

If $r = 0$, then p_n divides c, hence $n \in C$. If $r \neq 0$, then $n \notin C$. Since this is a contradiction with recursive inseparability of A and B, $+_M$ cannot be recursive. In order to show that $*_M$ is not recursive, code C by 2^c. This is possible, if we take c sufficiently small nonstandard. Then the procedure for testing $n \in C$ will search for $u, v \in M$, $u < v$ such that

$$M \models c = \underbrace{v * v * \cdots * v}_{p_n \text{ times}} * u ,$$

and decide according to the truth of $u = \overline{1}$. □

1.2 Theorem. Let M be a countable nonstandard model of $I\Sigma_0$. Then there exists a nonstandard initial segment of M which is a model of *PA*.

Proof. This is Theorem IV.3.39 strengthened form $I\Sigma_1$ to $I\Sigma_0$. Thus one only needs to check that the proof can be carried out in $I\Sigma_0$. We shall use a different argument. By Theorem IV.3.39, it suffices to prove that every countable model $M \models I\Sigma_0$ has a nonstandard initial segment which is a model of $I\Sigma_1$. We shall construct a semiregular short cut I and apply Theorem IV.2.16 to deduce that it is a model of $I\Sigma_1$. We have to use some properties of $I\Sigma_0$ which will be proved later in Sect. 3 (coding of sequences, definition of the graph of exponentiation).

Let M be a countable model of $I\Sigma_0$. Let $g(x, y)$ be defined as follows

$$g(0, y) = y^y ;$$
$$g(x + 1, y) = g^{y+1}(x, y) \quad (y + 1 \text{ iterated } \lambda x(g(x, y))) .$$

One can show that $g(x, y) = z$ can be defined by a Σ_0 formula and the inductive clauses can be proved in $I\Sigma_0$. Let $d \in M$ be a nonstandard number such that 2^{2^d} exists in M. By overspill we can find a nonstandard c such that $g(c, c)$ exists in M. Take an enumeration $f_0, f_1, \ldots$ of functions coded in M whose domain is $[0, c]$ such that each function occurs in it infinitely often. Define a nested sequence of intervals

$$[a_0, g(c, a_0)) \supseteq [a_1, g(c - 1, a_1)) \supseteq \ldots$$

with $a_0 = c$, as follows. Suppose the sequence has been defined up to $[a_i, g(c - i, a_i))$. Decompose this interval into

$$[a_i, g(c - i - 1, a_i)) \cup [g(c - i - 1, a_i), g^2(c - i - 1, a_i)) \cup \cdots$$
$$\cdots \cup [g^{a_i}(c - i - 1, a_i), g^{a_i+1}(c - i, a_i)) .$$

By the Pigeon Hole Principle (we can use it, since f_i is coded in M) one of the intervals in the subdivision does not intersect the range of f_i restricted to the domain $[0, a_i)$. This will be the next interval in the sequence. Let I be the cut determined by $a_0, a_1, \ldots$. The construction ensures that it is a cut (the successor function occurs infinitely often) it is semiregular (the range of f_i restricted to a_i is bounded by a_{i+1} in I), nonstandard (since $c \in I$) and short (since $I < d$). □

By the definition of Chap. 0 we allow only variables as the bounds in formulae of Σ_0. Let f be a function symbol. Extend L_0 by f and let us denote by Σ_0^f the class of bounded formulae in the extended language where we allow the bounds at the quantifiers to be *terms in the extended language* $L_0(f)$. Consider the following two theories:

$$I\Sigma_0^f(f) + \text{“}f \text{ is monotone”};$$
$$I\Sigma_0(f) + \text{“}f \text{ is monotone”}.$$

In the first one we extend the induction schema of $I\Sigma_0$ to Σ_0^f formulae, while in the second one we allow induction only for bounded formulae in $L_0(f)$ where the bounds at the quantifiers are only variables. We have already considered a special case with f equal to *exp*.

1.3 Proposition. The theories $I\Sigma_0^f(f) + \text{“}f$ is monotone” and $I\Sigma_0(f) + \text{“}f$ is monotone” are equivalent.

This proposition shows that it is not important which bounds are allowed at bounded quantifiers. The proof is almost identical with the proof of Lemma I.1.30.

It turns out that the *graphs* of exponentiation and several other important functions are definable by Σ_0 formulae. $I\Sigma_0$ does not prove that exponentiation (defined by such a formula) is a total function (see Theorem 1.4 below), but it does prove the inductive properties of it whenever the values exist. Thus instead of working in $I\Sigma_0(exp)$ (with the inductive properties for *exp*) we can take $I\Sigma_0$ with an axiom saying that *exp* defined by a Σ_0 formula is a total function. Then $I\Sigma_0(exp)$, and hence $I\Sigma_0^{exp}(exp)$, will be conservative extensions. (These extensions are conservative in a stronger sense: the new function symbol is definable in the subtheory.) The same holds for other monotone functions with Σ_0 definable graphs. This shows that extension of $I\Sigma_0$ by an axiom saying that a certain (Σ_0 definable monotone) function is total has the same effect as an extension of the language. Hence it is natural to consider such theories to be versions of the intuitive concept of *Bounded Arithmetic*.

It is useful to realize that $I\Sigma_0^f(f)$ has a $\Pi_1(f)$ axiomatization: replace the induction schema by the following version

$$(\forall x)(\varphi(\overline{0}) \,\&\, (\forall y < x)(\varphi(y) \to \varphi(S(y))) \to \varphi(x))$$

for $\Sigma_0(f)$ formulae and omit the axiom (Q3).

Let us recall some definitions from Chap. III.

(1) $|x| = \lceil \log_2(x+1) \rceil$;
(2) $\omega_0(x) = 2x$;
$\omega_{i+1}(x) = 2^{\omega_i(|x|-1)}$, for $x > 0$ and $\omega_{i+1}(0) = 0$;
(3) $2_0^x = x$,
$2_{i+1}^x = 2^{2_i^x}$.

Further we consider the following axioms:

(1) $\Omega_i \equiv_{df} (\forall x)(\exists y)(\text{“}y = \omega_{i+1}(x)\text{”})$;
(2) $Exp \equiv_{df} (\forall x)(\exists y)(\text{“}y = 2^x\text{”})$;
(3) $Superexp \equiv_{df} (\forall x)(\exists y)(\text{“}y = 2_x^x\text{”})$.

Here we use the quotation marks to denote a Σ_0 formula with the same meaning. At this point it is not obvious that such formulae exist. This will be proved in Sect. 3. In $I\Sigma_0$ we can also prove that ω_{i+1} grows faster than ω_i, 2^x grows faster than each ω_i and 2_x^x grows faster than 2^x. Hence if ω_{i+1} is total, then ω_i must be total too, etc. Thus we get the basic hierarchy of theories

$$I\Sigma_0 \subseteq I\Sigma_0 + \Omega_1 \subseteq I\Sigma_0 + \Omega_2 \subseteq \cdots \subseteq I\Sigma_0 + Exp \subseteq I\Sigma_0 + Superexp\,.$$

The most basic theorem about such systems of Bounded Arithmetic is the following one, which is usually referred to as *Parikh's Theorem.*

1.4 Theorem. Let ψ be a Σ_0^f formula and let π be a Π_0^f formula. Suppose that

$$I\Sigma_0^f(f) + \pi \vdash (\forall x)(f(x) \le f(x+1))\,,$$
$$I\Sigma_0^f(f) + \pi \vdash (\forall x)(\exists y)\psi(x,y)\,.$$

Then for some term $t(x)$ of $L(f)$

$$I\Sigma_0^f(f) + \pi \vdash (\forall x)(\exists y \le t(x))\psi(x,y)\,.$$

Proof. Suppose that the conclusion is false for any term t. Let c be a new constant. Thus for every term t, the following theory is consistent

$$I\Sigma_0^f(f) + \pi + (\forall y \le t(c))\neg\psi(c,y)\,.$$

Since we can bound any finite set of terms by a single one, (for example $I\Sigma_0^f(f) \vdash t_1(x), \ldots, t_n(x) \leq t_1(x) + \cdots + t_n(x)$), the following theory is also consistent

$$I\Sigma_0^f(f) + \pi + (\forall y \leq t_1(c))\neg\psi(c,y) + (\forall y \leq t_2(c))\neg\psi(c,y) + \cdots ;$$

here we use an enumeration of all terms of $L(f)$. Let M be a model of this theory and let K be defined by

$$K = \{a \in M \mid M \vDash a \leq t_i(c), \text{ for some } i\}.$$

Since f is monotone, K is a substructure of M. Since $I\Sigma_0^f(f) + \pi$ has a $\Pi_1(f)$ axiomatization K is a model of it. By the construction we have $K \vDash (\forall y)\neg\psi(c,y)$ which is a contradiction with the assumption of the theorem. □

The theorem can easily be generalized by relaxing the assumptions about the theory. The essential assumptions are that the theory has a $\Pi_1(f)$ axiomatization and proves some basic properties of terms. The theorem naturally transfers to the systems where f is not a function symbol but it is only defined by a bounded formula and an axiom saying that it is a total function is added.

Let us consider two important examples. For $I\Sigma_0$ this means that every Σ_0 definable provably total function is bounded by a *polynomial*. We shall code sequences (say of 0's and 1's) by binary expansions of numbers; the length of a sequence x is thus $|x|$. Hence any provably total function can increase the length of a sequence only *linearly*. This prevents us defining polynomial time computations in $I\Sigma_0$ and complicates also the formalization of other concepts. It can be easily seen that the axiom Ω_1 just captures the polynomial increase of the lengths. Simple computations yield that for every term t with ω_2 there exists a polynomial p such that

$$x \leq t(y) \to |x| \leq p(|y|)$$

and *vice versa*, for every polynomial p there exists a term with ω_2 such that

$$|x| \leq p(|y|) \to x \leq t(y).$$

Hence $I\Sigma_0 + \Omega_1$ is a theory in which it is more comfortable to work. In particular $I\Sigma_0 + \Omega_1$ is stronger than $I\Sigma_0$. More generally we have the following corollary.

1.5 Corollary. Let f and g be Σ_0 definable monotone functions. Suppose g grows faster than any function definable by a term in $L(f)$ (in the standard model). Let $\Omega(f)$, resp. $\Omega(g)$, be formulae saying that f, resp. g, is a total

function for some Σ_0 definition of the graph of f, resp. of g. Suppose that $I\Sigma_0 + \Omega(f)$ proves that f is monotone. Then $I\Sigma_0 + \Omega(f)$ does not prove $\Omega(g)$.

Proof. Let $\varphi(x,y)$ be the Σ_0 definition of $f(x) = y$ and let $\psi(x,y)$ be the definition of $g(x) = y$. (Hence e.g. $\Omega(g)$ is $(\forall x)(\exists y)\psi(x,y)$.) Let π be a formula in $L_0(f)$ defined by

$$\pi \equiv_{df} (\forall x,y)(\varphi(x,y) \equiv f(x) = y)\,.$$

Now suppose that $I\Sigma_0 + \Omega(f)$ does prove $\Omega(g)$. Then also

$$I\Sigma_0^f(f) + \pi \vdash \Omega(g)\,,$$

since π implies $\Omega(f)$. By Theorem 1.4

$$I\Sigma_0^f(f) + \pi \vdash (\forall x)(\exists y \le t(x))\psi(x,y)\,,$$

for some term in $L_0(f)$. But this is a contradiction with the assumption that g grows faster than any function definable by a term in $L_0(f)$. □

1.6 Corollary. The hierarchy

$$I\Sigma_0 \subseteq I\Sigma_0{+}\Omega_1 \subseteq I\Sigma_0{+}\Omega_2 \subseteq \cdots \subseteq I\Sigma_0 + Exp \subseteq I\Sigma_0 + Superexp$$

is proper. □

The sentences which separate the levels of this hierarchy are Π_2. We shall see in Sect. 5 that we can separate $I\Sigma_0{+}Exp$ from $I\Sigma_0{+}\Omega_n$ by a Π_1 sentence and the same is true for $I\Sigma_0 + Superexp$ and $I\Sigma_0 + Exp$. Such separations remain, however, an open problem for the theories below $I\Sigma_0 + Exp$.

*

Now we turn our attention to hierarchies of theories obtained by restricting the quantifier complexity of the formulae allowed in the schema of induction.

1.7 Definition. (1) E_0 is the class of open formulae in L_0;

(2) for $n > 0$, E_n is the class of bounded formulae in L_0 in the prenex normal form where the first quantifier is $\exists$, there are at most n alternations of quantifiers and the bounds at the quantifiers are terms in L_0 (which must not contain variables of the current and the following quantifiers).

Example. $(\exists x \le a)(\exists y \le a)(xy = a \,\&\, x > \overline{1} \,\&\, y > \overline{1})$ is an E_1 formula.

The theories IE_i are defined in the usual manner: Q plus the induction schema restricted to E_i formulae. Thus IE_0 is just I_{open}. There is a big gap between IE_0 and IE_1. In particular Theorems 1.1 and 1.2 hold for IE_1 and there are other reasons to consider IE_1 to be relatively strong. On the other hand we are not able to prove that IE_2 is stronger than IE_1; worse than that, we even do not know whether IE_1 is weaker than $I\Sigma_0$.

These questions are connected with difficult open problems in complexity theory and they are probably very difficult too. We can resolve same of these problems, if we assume some conjectures about complexity classes. This will be shown in Sect. 4. We shall use a different hierarchy of theories there. We shall extend language L_0 to a language L_2 by adding more function symbols, we shall use a stronger theory instead of Q and also we modify the definition of the classes of bounded formulae, (see Sect. 4). In this way we obtain a straightforward connection between the Polynomial Hierarchy (these complexity classes will be defined in the following section) and the hierarchy of bounded formulae.

The language L_2 contains a binary function symbol $\#$ whose intended interpretation is $2^{|x|*|y|}$. Other functions of L_2 can be bounded by polynomials. We have the following estimates

$$x \# y \leq \omega_2(4(x+y)^2) + 1\,;$$
$$\omega_2(x) \leq (x \# x)^4\,.$$

Hence each function defined by a term in L_2 can be estimated by a function defined by a term in $L_0(\omega_2)$ and *vice versa.* Thus if we use this language for defining a system of bounded arithmetic we should expect that we obtain something equivalent to $I\Sigma_0 + \Omega_1$. This system will be denoted by T_2. There is an equivalent system S_2 based on a modified schema of induction S_2. The classes in the hierarchy of bounded formulae in L_2 will be denoted by Σ_i^b, $i = 0, 1, \ldots$. The corresponding fragments will be denoted by T_2^i and S_2^i. Thus we have for instance: $IE_i \subseteq T_2^i$. We shall concentrate on subsystems of T_2 and S_2; we have mentioned the hierarchy IE_i only for sake of completeness.

* *

In the previous part of the book we have encountered several principles which can be used instead of induction. Let us mention here at least the most important one: *the least number principle.* We know that $L\Sigma_i$ (the least number principle for Σ_i formulae) is equivalent to $I\Sigma_i$ (induction for Σ_i formulae). We can expect that a similar situation occurs in Bounded Arithmetic, if we replace the Arithmetical Hierarchy by the hierarchy of bounded formulae. For most principles this is really so, for instance *induction* and *the least number principle* are equivalent also for the classes Σ_i^b, for $i \geq 1$.

In Bounded Arithmetic we have also some principles whose counterparts for Σ_i have not been considered, since they are trivially equivalent to induc-

tion. These are the schema *PIND*:

$$\varphi(\overline{0}) \,\&\, (\forall x)(\varphi(\llcorner x/2 \lrcorner) \to \varphi(x)) \to (\forall x)\varphi(x)\,,$$

and schema *LIND*

$$\varphi(\overline{0}) \,\&\, (\forall y < |x|)(\varphi(y) \to \varphi(S(y))) \to \varphi(|x|)\,.$$

The intuitive idea behind these schemata is the following. We can, in a sense, verify the conclusion $\varphi(a)$ using the inductive property of φ in *only* $|a|$ *steps.* Such a verification procedure is thus polynomial in *the size of input* a, while it is exponential for the ordinary induction. The schema *PIND* is used to define S_2 and its fragments S_2^i.

Another important schema is the *Pigeon Hole Principle*, see Chap. I, 2.20. For higher fragments, the strength of *PHP* is given by Theorem I.2.23. If we consider this principle for Σ_0 formulae, then it is at least as strong as $I\Sigma_0$. Whether it is derivable in $I\Sigma_0$ is an important open problem. We have some evidence that Σ_0-*PHP* might be stronger than $I\Sigma_0$; we can prove some mathematical results using Σ_0-*PHP* which we are not able to derive in $I\Sigma_0$ itself. In particular we have

$$Q + \Sigma_0\text{-}PHP \vdash (\forall x)(\exists p)(p > x \,\&\, \text{“}p \textit{ is a prime}\text{”})\,,$$

i.e. Σ_0-*PHP* proves that there are infinitely many primes, while we are not currently able to derive it in $I\Sigma_0$. If Σ_0-*PHP* *is* stronger than $I\Sigma_0$ then a philosophical question arises: *what is a more basic principle, Induction or the Pigeon Hole Principle*?

* * *

We have not exhausted all weak fragments that have ever been considered. You can find more in the literature listed in *Bibliographical Remarks and Further Reading.*

2. A Brief Introduction to Complexity Theory

We shall outline some basic concepts of complexity theory. We shall present a little more than is needed for understanding the following sections, but we shall prove only those results which are in some sense connected with the following text. From the point of view of logic, the most interesting subject in complexity is the quantifier hierarchies such as the Polynomial Hierarchy.

(a) Time and Space Complexity Classes

There are two basic computational models in complexity theory: Turing machines and Boolean circuits. Here we consider only the first one, and also we shall use only one type of it.

By a *Turing machine* we mean *an off-line multitape Turing machine*, which is a finite automaton connected with a finite number of infinite (say in one direction only) tapes. The automaton is also called *a finite control*. The tapes consist of cells on which a symbol of a finite alphabet can be written. The automaton operates heads on tapes. A head can read and rewrite the symbols and move to the next cell (left or right). One of the tapes is distinguished as *the input tape*; the automaton cannot write on this tape. The automaton has a distinguished initial state and a subset of final states. A more up-to-date description would be in terms of a very restricted programming language (instead of the finite automaton) operating with linear arrays of symbols (instead of tapes).

A Turing machine is used to define a language (i.e. a set of strings in a finite alphabet) or a mapping from the set of strings in an alphabet to another such set. The first type is called an acceptor, the second type is called a transducer. An *acceptor* is a Turing machine which has an input tape and several additional tapes called *work tapes*, and it has a distinguished subset of final states called *accepting states*. A word w is accepted if the machine reaches the accepting state when started in the initial state with w on the beginning of the input tape and with the rest of the tapes blank. The language *accepted by* the machine consists of all words accepted by it. A *transducer* is a similar device except that it has one more distinguished write-only tape. It computes a partial function F in an obvious way: to compute $F(w)$, start with w on the input tape and read $F(w)$ on the output tape when it reaches a final state.

In one step, the machine reads the tapes, rewrites them, moves the heads by one cell and changes the state of the finite control. The *time* of the computation on w is the number of steps needed to reach an accepting state. The *space* is the number of cells used on the work tapes.

At first it may seem that Turing machine time is a very bad approximation of the time needed by actual computers, because the actions of a Turing machine are very restricted. It turns out, however, that Turing machines are quite time efficient due to the fact that they have several tapes. In the case of the space complexity measure this is even more true and having more than one work tape is no advantage.

2.1 Definition. Let $f : N \to N$ be a function. We say that a language L is *accepted in time* (resp. *space*) $f(n)$, if there exists a Turing machine M which accepts L and such that for every n and any w, $|w| = n$, the time (resp.

space) of M on w is at most $f(n)$. We define

$$\mathit{Time}(f(n)) = \{L \mid L \text{ accepted in time } f(n)\},$$
$$\mathit{Space}(f(n)) = \{L \mid L \text{ accepted in space } f(n)\}.$$

Note that we do not fix the alphabet on work tapes. Hence if we increase the size of the alphabets we can obtain a faster and more space efficient machine. This is the idea behind the following theorem.

2.2 Theorem. (a) For $f(n)$ such that $\lim f(n)/n = \infty$ and $c > 0$,

$$\mathit{Time}(\max(n, f(n))) = \mathit{Time}(\max(n, cf(n))).$$

(b) For any $f(n)$ and any $c > 0$,

$$\mathit{Space}(f(n)) = \mathit{Space}(cf(n)).$$

Interesting complexity classes are obtained by taking the unions over natural classes of bounding functions.

2.3 Definition.

$\mathit{LinTime} = \bigcup_{c\in N} \mathit{Time}(cn)$	(Linear Time)
$P = \bigcup_{c\in N} \mathit{Time}(n^c)$	(Polynomial Time)
$E = \bigcup_{c\in N} \mathit{Time}(c^n)$	
$\mathit{Exp} = \bigcup_{c\in N} \mathit{Time}(2^{n^c})$	(Exponential Time)
$\mathit{LogSpace} = \mathit{Space}(\log n)$	(Logarithmic Space)
$\mathit{LinSpace} = \mathit{Space}(n)$	(Linear Space)
$\mathit{PSpace} = \bigcup_{c\in N} \mathit{Space}(n^c)$	(Polynomial Space)

Note that by Theorem 2.2 we have $\mathit{LogSpace} = \bigcup_{c\in N} \mathit{Space}(c.\log n)$ and $\mathit{LinSpace} = \bigcup_{c\in N} \mathit{Space}(c.n)$.

We have the following trivial inclusions

$$\mathit{LinTime} \subseteq P \subseteq E \subseteq \mathit{Exp}$$
$$\mathit{LogSpace} \subseteq \mathit{LinSpace} \subseteq \mathit{PSpace}$$

Below we shall explain why these inclusions are in fact strict. We have also trivially $\mathit{Time}(f(n)) \subseteq \mathit{Space}(f(n))$, but there is a nontrivial result which gives more:

2.4 Theorem. For $f(n) \geq n$, $\mathit{Time}(f(n)) \subseteq \mathit{Space}(f(n)/\log f(n))$.

In the other direction only the trivial inclusion

$$Space(f(n)) \subseteq \bigcup_{c \in N} Time(c^{f(n)})$$

is known. (An $f(n)$ space bounded machine cannot compute in time longer than $c^{f(n)}$, because it would run into a cycle.) Thus for instance, $LogSpace \subseteq P$, but it is a famous problem whether the inclusion is strict.

The complexity classes above are defined for languages. We shall talk about the complexity of *subsets of nonnegative integers* using the standard binary coding. We can talk also about the complexity of relations, since we can code k-tuples of words by single words using an additional symbol for separating them. Using transducers instead of acceptors we can define corresponding classes of *functions*. (Note that there is no simple reduction of a function class to the corresponding language class.) We shall not need the function classes corresponding to the classes of languages defined above except for the function class corresponding to P; it will be denoted by $\square_1^p$.

Let us mention an alternative notation: the subsets of natural numbers which belong to $LinSpace$ are also known as Grzegorczyk's class E_*^2.

(b) Nondeterministic Computations

A nondeterministic Turing machine is a Turing machine in which the finite automaton is nondeterministic (above we have considered only deterministic machines). This means that in some states there is more than one possible action of the machine. Thus there might be many possible computations on a given input word w. A word w is *accepted* by a machine if there exists (at least one) computation leading to an accepting state. When counting the computation resources needed for a given input word (time and space) we consider the worst case. Now we can define nondeterministic counterparts of the deterministic classes. They will be denoted simply by prefixing N. For example:

$$NP = \bigcup_{c \in N} NTime(n^c) \quad \text{(Nondeterministic Polynomial Time)}.$$

Nondeterminism adds much power to the computations. We can *guess* a solution to a problem and then *check* its correctness. This suggests the following, more logical, definition of NP:

$$L \in NP \equiv (\exists R(x,y) \in P)(\exists p\text{-polynomial}) L = \{x \mid (\exists y, |y| \leq p(|x|)) R(x,y)\},$$

see Theorem 2.11 below.

As an example consider the complexity of the set of composite numbers. We can guess a proper divisor of a number k and check that it really is a

divisor of k. This proves that this set is in NP; it is an open problem whether it is in P.

The most important problem in complexity theory (and one of the most important problems in contemporary mathematics) is the question whether $P = NP$. It is a part of a more general problem about the relation of nondeterministic classes to deterministic ones. We have the following trivial inclusions

$$Time(f(n)) \subseteq NTime(f(n)) \subseteq Space(f(n)),$$

but for converse simulations we have only exponential bounds. For space classes the situation is a little different:

2.5 Theorem. $NSpace(f(n)) \subseteq Space(f(n)^2)$, provided $f(n)$ can be computed in space $f(n)$. Hence $NPSpace = PSpace$.

The next natural step is to consider the role of the universal quantifier. Traditionally this is done by taking complements of the nondeterministic classes. Thus we introduce another prefix for complexity classes, *co*, to denote the class of the complements of the sets in a given class. (It is superfluous in connection with deterministic classes, since they are closed under complements.) The problem whether $NP = coNP$ arises quite often in connection with problems in logic. It seems likely that the time classes are *not* closed under complements. On the other hand, quite surprisingly, a proof has been found recently showing that nondeterministic space classes *are* closed under complements.

2.6 Theorem. $coNSpace(f(n)) = NSpace(f(n))$, provided $f(n) \geq \log_2 n$ can be computed in nondeterministic space $f(n)$.

However, it is still open whether $Space(f(n)) = NSpace(f(n))$, in particular whether $LogSpace = NLogSpace$.

(c) Degrees and *NP*-completeness

A set (language) A is *polynomially reducible* to a set B if there exists a polynomial time computable function F (i.e. $F \in \square_1^p$) such that $F^{-1}(B) = A$. Clearly, this relation is a quasiordering, hence defines a structure of degrees. The smallest degree is P.

2.7 Theorem. There exists the largest degree contained in NP.

The sets in this degree are called *NP-complete*. Not only do NP-complete sets exist, but they occur very naturally. There are hundreds of naturally defined NP-complete problems. This fact is important for the $P \stackrel{?}{=} NP$ problem:

we know that *NP*-complete sets are *not* in P, if $P \neq NP$ (which is accepted by most people almost as an axiom). Another consequence is that the statement "$P \neq NP$" is equivalent to a Π_2 formula.

Examples of *NP*-compete Problems.

(1) *CLIQUE* – the set of pairs (G, k) where G is a graph containing a clique (i.e. a complete subgraph) of size k. (We assume that graphs are suitably coded as strings in a finite alphabet.) This is a typical *NP*-complete set, since it is connected with an optimization problem: find the largest clique in a given graph. We shall talk about optimization problems at the end of Sect. 4.

(2) *HAMILTONIAN GRAPHS* – the set of graphs that contain a cycle containing all vertices.

(3) The set of triples (a, b, c) of integers $a, b, c > 0$ such that the equation

$$ax^2 + by = c \tag{c.1}$$

is solvable in natural numbers.

The last example shows that there are *NP*-complete problems expressible by bounded formulae, since, clearly, we can bound any solution x, y to (c.1) by c. This sets some limitations on the provability of Matiyasevič's theorem in weak fragments. Consider the formula

$$\varphi(a, b, c) \equiv_{df} (\forall x, y \leq c)(ax^2 + by \neq c).$$

By Matiyasevič's theorem there exists an open formula $\psi(a, b, c, \mathbf{z})$ such that

$$N \models \varphi(a, b, c) \equiv (\exists \mathbf{z})\psi(a, b, c, \mathbf{z}),$$

where $\mathbf{z} = z_1, \ldots, z_k$. Suppose this equivalence was provable in $I\Sigma_0 + \Omega_1$. Then in particular,

$$I\Sigma_0 + \Omega_1 \vdash \varphi(a, b, c) \to (\exists \mathbf{z})\psi(a, b, c, \mathbf{z}).$$

Hence, by Theorem 1.4, we can add bounds to the variables $\mathbf{z}$ with the following form

$$|z_1|, \ldots, |z_k| \leq p(|a|, |b|, |c|),$$

where p is a polynomial. Thus $\varphi(a, b, c)$ would be equivalent to

$$(\exists \mathbf{z}, |z_1|, \ldots, |z_k| \leq p(|a|, |b|, |c|))\psi(a, b, c, \mathbf{z}),$$

which would prove that $\varphi(a, b, c)$ defines a set in *NP*. It can easily be proved that if the complement of an *NP*-complete set is in *NP* then $NP = coNP$.

Hence if $NP \neq coNP$, then the bounded formula $\varphi(a, b, c)$ is not equivalent to an existential formula in $I\Sigma_0 + \Omega_1$.

Let us mention only one more result about the degrees.

2.8 Theorem. If $P \neq NP$, then the degrees below NP are dense; in particular, there is a degree strictly between P and NP-complete sets.

(d) Oracle Computations

An *oracle Turing machine* is a Turing machine with a special tape, called *an oracle tape*, and special *oracle states*. In an oracle state the machine can perform exactly two actions. Given a set A, *an oracle*, the machine computes in the usual manner except when it reaches an oracle state. Then it asks whether the current word on the oracle tape belongs to A and acts according to the answer. Thus if the non-oracle states are deterministic, the computation is deterministic (provided A is fixed). In order to limit the amount of information provided by an oracle in the case of linear time computations, we shall assume that before the next query is asked, the previous one is erased.

Let C be a complexity class and let A be a set (an oracle). We denote by $C(A)$ the class defined in the same way as C except that the Turing machines are augmented with the oracle A. We say that $C(A)$ is C *relativized to* A. We can also take a class $\mathcal{A}$ of oracles, in particular a complexity class, and consider

$$C(\mathcal{A}) =_{df} \bigcup_{A \in \mathcal{A}} C(A).$$

Almost all statements about relations between complexity classes relativize, which means that they remain true if we add an oracle. On the other hand there is the following result which shows that the $P \stackrel{?}{=} NP$ problem cannot be solved without going deep into the combinatorial structure of polynomial time computations.

2.9 Theorem. There are oracles A an B such that $P(A) = NP(A)$ and $P(B) \neq NP(B)$.

Similar "independence" can be shown for other open problems about complexity classes.

(e) The Linear Time Hierarchy and the Polynomial Hierarchy

2.10 Definition.

$\Sigma_0^{lin} =_{df} LinTime$;

$\Sigma_{i+1}^{lin} =_{df} NLinTime(\Sigma_i^{lin})$;

$LinH =_{df} \bigcup_i \Sigma_i^{lin}$ (*Linear Time Hierarchy*);

$\Sigma_0^p =_{df} P$;

$\Delta_{i+1}^p =_{df} P(\Sigma_i^p)$;

$\Sigma_{i+1}^p =_{df} NP(\Sigma_i^p)$;

$PH =_{df} \bigcup_i \Sigma_i^p$ (*Polynomial (Time) Hierarchy*);

$\square_1^p$ – functions computable in polynomial time;

$\square_{i+1}^p =_{df} \square_1^p(\Sigma_i^p)$ – functions computable in polynomial time with oracles from Σ_i^p.

In particular, we have $\Sigma_1^{lin} = NLinTime$ and $\Sigma_1^p = NP$. As usual we write Π_i^p for $co\Sigma_i^p$ and similarly for the classes of the Linear Time Hierarchy. We have also the same diagram of inclusions as for the Arithmetic Hierarchy, but it is an open problem whether any of these inclusions is strict, both for the Polynomial Hierarchy and for the Linear Time Hierarchy.

We shall show that the "logical" definition of NP extends to all classes of the Polynomial Hierarchy and that there is a similar representation for the classes of the Linear Time Hierarchy. In order to simplify the formulae, let us write

$$(\exists^{lin} x)\varphi \text{ for } (\exists x, |x| \leq \ell(|y_1|, \ldots, |y_i|))\varphi\,,$$

where $\ell(|y_1|, \ldots, |y_i|)$ is a linear function (with positive coefficients) and $y_1, \ldots, y_i$ are the free variables of φ. $\forall^{lin}$ has the dual meaning; $\exists^p$ and $\forall^p$ are defined in the same way except that we use polynomials instead of linear functions.

2.11 Theorem. (a) Σ_i^{lin} is the class of sets defined by formulae of the form

$$(\exists^{lin} x_1)(\forall^{lin} x_2)\ldots(Q^{lin} x_i)R(w, x_1, x_2, \ldots, x_i)\,,$$

where R is in $LinTime$, and Q is $\exists$ if i is odd and $\forall$ if i is even.

(b) Σ_i^p is the class of sets defined by formulae of the form

$$(\exists^p x_1)(\forall^p x_2)\ldots(Q^p x_i)R(w, x_1, x_2, \ldots, x_i)\,,$$

where R is in P and Q is $\exists$ if i is odd and $\forall$ if i is even.

Proof. We shall sketch a proof of (a), since (b) is similar and, in fact, simpler. For $i = 0$ it is trivial. Consider $i = 1$. Thus we have a nondeterministic Turing

machine M running in linear time and an oracle for a set in *LinTime*. We would like to express the set accepted by the oracle machine by a formula $(\exists^{lin} x_1)R(w, x_1)$ with R in *LinTime*. We shall use x_1 to code the following information:

(1) the nondeterministic decisions of the machine M,
(2) the queries asked by the machine, and
(3) the answers of the oracle.

R will be computed by a machine which simulates M *deterministically* using x_1 and using the subroutine for the oracle. Note that the total time needed to run the subroutines is linear on the input size, since the sum of the lengths of the oracle queries is linear.

Consider a general i. We use x_1 to code the same information. By the induction assumption, the oracle answers are determined by a formula

$$(\exists^{lin} x'_1)(\forall^{lin} x'_2)\dots(Q^{lin} x'_{i-1})R'(w', x'_1, x'_2, \dots, x'_{i-1})\,. \tag{e.1}$$

We need to express that the oracle answers coded by x_1 are correct. This can be done by a conjunction of instances of (e.1) and its negation where w' runs through the oracle answers. The negations of (e.1) are equivalent to formulae starting with a universal quantifier, hence their conjunction can be easily expressed as a single formula of the form

$$(\forall^{lin} x''_1)(\exists^{lin} x''_2)\dots(Q^{lin} x''_{i-1})R''(w'', x''_1, x''_2, \dots, x''_{i-1})\,.$$

The nonnegated instances of the formula (e.1) can also be expressed by a single formula. Again the trick is that the sum of the lengths of the oracle queries is linear, hence also the sum of the lengths of the variables x'_1 can be bounded using a linear function. We omit the details. The converse implication is easy. □

We shall show that *LinH* consists of sets definable by Σ_0 formulae (in the language L_0; we consider only subsets of natural numbers). Later, in Sect. 4, we shall show that *PH* consists of sets definable by bounded formulae in L_2. That is why these complexity classes are so important for us. We shall denote by Σ_0^N the subsets of natural numbers definable by Σ_0 formulae.

2.12 Lemma. $\mathit{LinTime} \subseteq \Sigma_0^N$.

Proof. We shall assume that a suitable coding of sequences can be expressed using Σ_0 formulae. This will be done with details in Sect. 3. (Note that our task in Sect. 3 will be harder, since we shall develop coding of sequences in the *theory* $I\Sigma_0$.) Furthermore we need the function *nuon* which counts the number of ones in a binary expansion of a number. A Σ_0 formula defining this function is also constructed in Sect. 3.

Our goal is to express, by a Σ_0 formula $\varphi(w)$, that a machine M accepts a word w provided that M runs in linear time. To this end we need to code the computation of M by a sequence s whose length is linear in the length of w. This is because the polynomial bound on the size of numbers, as it is in Σ_0, imposes a linear bound on their length and *vice versa*. We code sequences essentially by bits in binary expansions (see Sect. 3 for details).

Suppose M has k heads. The sequence s will consist of $(k+1)$-tuples, one for each computation step. The first member of the $(k+1)$-tuple will code the state of M. The remaining members will be pairs, one for each head. They will code the currently read symbol and the direction in which the head will move. Clearly the length of (the code of) s is linear in the length of w. The sequence s does not contain information about the positions of heads on tapes explicitly, but this can be defined from s by a Σ_0 formula. The position of a head in a computation step is the number of its moves to the right minus the number of its moves to the left prior to this step. The number of moves to the right (similarly to the left) is the number of occurrences of the symbol coding these moves in s. This can be easily reduced to the function *nuon*. Now for a given head and a given computation step (as coded in s), we can determine using Σ_0 formulae whether the currently read cell has been visited before, and if it has, then we can determine the last visit.

In order to state that s is an accepting computation, we have to impose conditions on $(k+2)$-tuples

$$c, c^+, a, c_2, \ldots, c_k ,$$

where c is en element of s, c^+ is its successor, a is the symbol in the input word w read in the step corresponding to c and $c_2, \ldots, c_k$ are elements of s which correspond to the last visits of work heads of the cells read in c, (some of them may be missing if some heads visit some cells for the first time). Since the automaton of M is finite and all the alphabets are finite, there are finitely many possibilities for such a $(k+2)$-tuple and they can be listed. Finally, of course, we add that the last state is an accepting state. □

2.13 Lemma. $LinH \subseteq \Sigma_0^N$.

Proof. This follows from Lemma 2.12, Theorem 2.11 and the observation made above that the growth rate of bounds in Theorem 2.11 and those of Σ_0 are the same. □

Before showing the converse inclusion, we prove an important result.

(f) Nepomnjaščij's Theorem

We need to introduce another type of complexity class, classes defined by simultaneous resource bounds:

TimeSpace$(f(n), g(n))$ is the class of sets which can be accepted by Turing machines running *simultaneously* in time $f(n)$ and space $g(n)$. Note that this class is in general smaller than *Time*$(f(n)) \cap$ *Space*$(g(n))$.

2.14 Theorem. Let ε be a rational number, $0 < \varepsilon < 1$, and let a be a positive integer. Then *TimeSpace*$(n^a, n^\varepsilon) \subseteq$ *LinH*.

Proof. First we shall sketch an idea of the proof in which we shall assume that the numbers n^α needed in the proof, with α rational, are integers and that they are computable in time n^α (which is not always true).

Let ε, $0 < \varepsilon < 1$, be given. We shall prove by induction on k that *TimeSpace*$(n^{k(1-\varepsilon)}, n^\varepsilon) \subseteq$ *LinH*. For $k = 1$ it is trivial. Suppose it holds for k. Let M be a Turing machine. We shall call *a configuration* the complete description of the machine state in a computation step, i.e. the state of the finite control, the position of heads, the content of the work tapes; only the input word is not included. We assume that configurations are naturally encoded as strings over a finite alphabet. Suppose M runs in space n^ε. Let an input word w and two configurations c_1, c_2 of M be given. Then we can test in *TimeSpace*$(n^{k(1-\varepsilon)}, n^\varepsilon)$ whether M will reach c_2 from c_1 in $n^{k(1-\varepsilon)}$ steps: just simulate M. Hence, by the induction hypothesis, this condition on w, c_1, c_2 can be tested in *LinH*. Now let M' be a machine running in time $n^{(k+1)(1-\varepsilon)}$ and space n^ε. To show that M' accepts a set in *LinH* we shall simulate M' by an oracle nondeterministic Turing machine K running in linear time. Let w be an input word. First, K constructs nondeterministically $n^{1-\varepsilon} + 1$ strings. Then it checks that:

(1) the first string is the initial configuration of M';
(2) every pair of consecutive strings are configurations of M' such that M' can reach the second one from the first one in $n^{k(1-\varepsilon)}$ steps;
(3) the final string is an accepting configuration.

Here (1) and (3) can be done deterministically in linear time (always assuming a reasonable coding of configurations). For (2) it will use the oracle shown to exist above. We need only to show that K runs in linear time. The key idea of the whole proof is that the total length of the $n^{1-\varepsilon} + 1$ configurations is linear in n. This is because M' runs in space n^ε; hence the strings representing configurations of M' have lengths linear in n^ε.

We have omitted several details, but we shall only comment on the one mentioned above. The problem is how to compute $\ulcorner n^{k(1-\varepsilon)} \urcorner$. The point is that, due to nondeterminism, we do not have to count the number of steps during the simulation at all. More precisely, in the inductive construction we need only a relation $R(w, c_1, c_2)$ in *LinH* such that

(1) if M will reach c_2 from c_1 in $n^{k(1-\varepsilon)}$ steps on input w then $R(w, c_1, c_2)$, and

(2) if $R(w, c_1, c_2)$ then M will reach c_2 from c_1 on input w (using possibly even more than $n^{k(1-\epsilon)}$ steps).

The simulating machine can guess the intermediate configurations that are evenly (as much as possible) distributed. □

We shall use this theorem directly only in a very special case (the next corollary). However the *idea* of using quantifiers in order to code longer than linear computations is very useful. We shall use it in Sect. 3 to construct a Σ_0 definition of the function *nuon*.

2.15 Corollary. *LogSpace* $\subseteq$ *LinH*.

Proof. We know that *LogSpace* $\subseteq P$, but the proof of this fact gives more:

$$\mathit{LogSpace} \subseteq \bigcup_k \mathit{TimeSpace}(n^k, \log n)\,.$$

Hence, by Theorem 2.14, *LogSpace* $\subseteq$ *LinH*. □

2.16 Theorem. Σ_0 definable sets are just the sets in the Linear Time Hierarchy, i.e. $\Sigma_0^N = \mathit{LinH}$.

Proof. By Lemma 2.12, it remains to prove $\Sigma_0^N \subseteq \mathit{LinH}$. By Theorem 2.11 and since the bounded quantification is in both cases of the same growth rate, we only need to show that the open arithmetical formulae define sets in *LinH*. It can be easily shown that *LinH* is closed under set operations, hence the question reduces to atomic formulae. The relations $=$ and $\leq$ are easily computable in linear time and so is addition. We do not know whether multiplication of two numbers is computable in linear time (probably it is not), but we can show that it is computable in logarithmic space. This can be proved by analyzing the school algorithm for multiplication. Suppose we multiply a number of length n with a number of length m in binary. First we produce m numbers of length at most $n+m$ and then we add them. This needs space of the order mn. However to add these numbers we do not have to write them down. We only have to remember the current row and the column and we can always compute the entry at this place. To remember the position, we need about $\log m + \log n$ bits. Furthermore, we have to remember the carry, which is of length at most $\log m$. Thus the total space needed to compute the product is logarithmic in the length of the numbers. It is also easily seen that we do not need to compute the multiplication as an operation; we only need to compute its graph. Hence, by Corollary 2.15 the graph of multiplication is in *LinH*. □

(g) The Diagonal Method for Separating Complexity Classes

The most important problems in complexity theory can be stated as problems about separating some complexity classes (which means showing that they are different). In Bounded Arithmetic the most important problems are also about separating fragments of Bounded Arithmetic. There are two kinds of such problems. The first kind concerns fragments defined by restricting the quantifier complexity of an induction schema. For instance we do not know whether $IE_i = IE_{i+1}$, $S_2^i = S_2^{i+1}$ and $T_2^i = T_2^{i+1}$. In the problems of the second kind, the meaning of "separation" is Π_1 nonconservation. The typical problem here is whether $I\Sigma_0 + \Omega_1$ is Π_1 conservative over $I\Sigma_0$. For each fragment of Bounded Arithmetic there is a naturally assigned complexity class: the class of sets which are definable by formulae for which the induction schema is provable in the fragment. Thus Σ_0^N, i.e. $LinH$, corresponds to $I\Sigma_0$, PH corresponds to $I\Sigma_0 + \Omega_1$, etc. The separation problems for fragments and those for the corresponding complexity classes seem to be connected, though we know very little about it (a connection is shown at the end of Sect. 4). (Let us note that one can consider different schemas of induction and different base theories, thus the correspondence between fragments and complexity classes is not one-to-one.)

Here we would like to discuss a method on which almost all separation results in complexity rely. It is called *diagonalization*. Its applicability is limited; a rule of thumb is that we can separate only complexity classes defined using bounds to the same resources. The method turns out to be useless for time-space and determinism-nondeterminism problems. A natural question to ask now is what corresponds to this method in fragments of arithmetic. In metamathematics of arithmetic diagonalization is well-known: it is the essence of Gödel's theorems. It seems that it has similar limitations in fragments of Bounded Arithmetic. In particular, no fragments have been separated for which the corresponding problem in complexity theory is open.

We need some technical concepts before stating the basic results which use diagonalization. A function f is *space constructible* if there exists a Turing machine with one work tape which on each standard input of length n (say a word consisting of n 0's) uses a space of exactly $f(n)$. A function f is *time constructible* if there is a Turing machine which uses a time of exactly $f(n)$ on each standard input of length n. These concepts are needed to avoid having possible pathological functions as space or time bounds; the usual bounding functions are easily shown to be space and time constructible.

2.17 Theorem. (i) Let f and g be space constructible, $f(n) \geq \log_2 n$ and suppose

$$\liminf f(n)/g(n) = 0\,.$$

Then

$$Space(g(n)) \setminus Space(f(n)) \neq \emptyset\,.$$

(ii) Let f and g be time constructible and suppose

$$\liminf f(n)/g(n) = 0\,.$$

Then

$$\mathit{Time}(g(n)\log_2 g(n))\setminus \mathit{Time}(f(n)) \neq \emptyset\,.$$

Thus we have $\mathit{LinTime} \subsetneqq P \subsetneqq E \subsetneqq \mathit{Exp}$ and $\mathit{LogSpace} \subsetneqq \mathit{LinSpace} \subsetneqq \mathit{PSpace}$. An interesting consequence of $\mathit{LogSpace} \subsetneqq \mathit{PSpace}$ is that at least one of the inclusions in the following sequence is strict:

$$\mathit{LogSpace} \subseteq P \subseteq \mathit{NP} \subseteq \mathit{PSpace}\,.$$

But for each of them it is an open problem.

The idea of the **proofs** of both statements is essentially the same. Take a suitable coding of Turing machines as strings in an alphabet Σ. Let $\#$ be a symbol not in Σ. Construct a Turing machine M which works as follows. On each input it will simulate a machine computing the bound $g(n)$ and stop if the bound is achieved, so M is space (resp. time) bounded by g. If an input w is a code of a machine K followed by a string of $\#$'s, at the same time (say, on an extra tape) M will simulate K on w in space (resp. time) limit $g(|w|)$. If the simulation of K is finished before M runs out of space (resp. time), then M accepts w iff K does not. This situation must happen for each K which has a space (resp. time) bound $f(n)$, since the bound $g(n)$ for M is larger for a sufficiently long w. Thus the language accepted by M is different from the language accepted by each such K. The reason for $g(n)\log_2 g(n)$ instead of just $g(n)$ in (ii) is that we have to simulate by M machines with an arbitrary number of tapes and this is not possible without an increase in time.

The proof of the corresponding results for nondeterministic classes is more difficult. The problem is how to arrange that M accepts the input iff K does not. In the case of nondeterministic space classes we could use the fact that they are closed under complements (Theorem 2.6), but this cannot be used in the case of nondeterministic time classes. There is, however, a nice trick for avoiding this problem. This trick works even in more general situations.

2.18 Theorem. Let C be a class of languages in an alphabet Σ, with $0 \in \Sigma$, let L be a language in the alphabet Σ, let r be a mapping assigning a word in the alphabet Σ to each language in C and let z be a mapping from C to natural numbers. Suppose that for every $K \in C$

(a) $r(K) \frown 0^{z(K)} \in L \equiv r(K) \notin K$,

(b) $(\forall j, 0 \leq j < z(K))(r(K) \frown 0^j \in L \equiv r(K) \frown 0^{j+1} \in K)$.

Then $L \notin C$. (0^j denotes the string of j 0's.)

Proof. Suppose $L = K$ for some $K \in C$. Then

$$r(K) \frown 0^{j+1} \in L \equiv r(K) \frown 0^{j+1} \in K\,,$$

hence by (b),

$$r(K) \in L \equiv r(K) \frown 0^{z(K)} \in L\,.$$

Thus by (a)

$$r(K) \in L \equiv r(K) \notin K\,,$$

which is a contradiction. □

In the usual diagonalization we have only the simple condition $r(K) \in L \equiv r(K) \notin K$, which is, however, sometimes more difficult to satisfy than (a) and (b) above. This is because we can take z to be an exponentially growing function. Then in order to determine whether $r(K) \frown 0^{z(K)} \in L$, we need to decide whether a much smaller word $r(K)$ belongs to K. Theorem 2.18 can be used not only for nondeterministic classes, but also for classes with a larger quantifier prefix. We shall show only a very special case of such a theorem.

2.19 Theorem. For every $i \geq 0$, $\Sigma_i^{lin} \subsetneq \Sigma_i^p$.

Proof. Let i be given. We shall apply Theorem 2.18 with $C = \Sigma_i^{lin}$. We shall also use the representation of Σ_i^{lin} and Σ_i^p given by Theorem 2.11. Thus every $K \in \Sigma_i^{lin}$ is defined by a formula

$$(\exists^{lin} x_1)(\forall^{lin} x_2)\ldots(Q^{lin} x_i)R(w, x_1, x_2, \ldots, x_i)\,, \tag{g.1}$$

where R is in *LinTime*. The diagonal language L will be defined by a formula of the form

$$(\exists x_1, |x_1| \leq |w|^2)(\forall x_2, |x_2| \leq |w|^2)\ldots(Q x_i, |x_i| \leq |w|^2)S(w, x_1, x_2, \ldots, x_i)\,,$$

with $S(w, x_1, x_2, \ldots, x_i)$ computable in time $|w|^4$, for $|x_1|, \ldots, |x_i| \leq |w|^2$, by a two tape deterministic Turing machine. We shall use the result that an $f(n)$ time bounded multitape Turing machine can be simulated by a $cf(n)\log_2 f(n)$ time-bounded two tape Turing machine, with c a constant. (We do not need such a good simulation; a simple polynomial simulation is sufficient, but then $|w|^4$ must be replaced by a polynomial of higher degree.) Thus in order to simulate a linear time Turing machine, we need asymptotically less than n^2 time.

Let $K \in \Sigma_i^{lin}$ be defined by a formula (g.1) with $\ell_1(|w|), \ldots, \ell_i(|w|)$ as the implicit linear bounds for the quantifiers. We take $r(K)$ to be a word which contains a code of the machine computing R and the codes of the

linear bounds, and moreover assume it is sufficiently long. Namely we want, for $n \geq |r(K)|$,

$$\ell_1(n+1) \leq n^2, \ldots, \ell_i(n+1) \leq n^2,$$

and that a computation of the predicate

$$|x_1| \leq \ell_1(|w|) \,\&\, ((|x_2| \leq \ell_2(|w|) \rightarrow (|x_3| \leq \ell_3(|w|) \,\&\, (\ldots \\ \ldots R(w, x_1, x_2, \ldots, x_i) \ldots),$$

for $|x_1|, \ldots, |x_i| \leq |w|^2$, $|w| \geq n$ be possible in time $(|w|-1)^4$. Furthermore we construct $r(K)$ so that it does not contain occurrences of 0. Clearly each such a K can be computed in exponential time. There is a fixed exponential bound $f(n)$ such that each such K can be computed within this bound by a two tape Turing machine *for sufficiently long inputs*. Again, sufficiently long means longer than $|r(K)|$. We define $z(K) = f(|r(K)|)$.

Now we define the machine M which determines the predicate S. Given an input $(w \frown 0^j, x_1, x_2, \ldots, x_i)$, where the last symbol of w is not 0, M will compare j and $f(|w|)$. If $j > f(|w|)$ then what it does is not important. If $j = f(|w|)$, then M will try to interpret w as a code of a machine and linear bounds which determine a language K defined by a formula of the form (g.1). Again it is important only to consider the case where w is such a code. Then M will (deterministically) decide whether $w \in K$. This is possible in time $|w \frown 0^j|^2$, since $j = f(|w|)$. M accepts such an input iff $w \notin K$. (Thus in this case $x_1, \ldots, x_i$ are not used at all.) Clearly this ensures condition (a) of Theorem 2.18. Now suppose that $j < f(|w|)$. In this case M will also try to interpret w as a code of a machine and linear bounds, but now it will compute the predicate

$$|x_1| \leq \ell_1(|u|) \,\&\, ((|x_2| \leq \ell_2(|u|) \rightarrow (|x_3| \leq \ell_3(|u|) \,\&\, (\ldots \\ \ldots R(u, x_1, x_2, \ldots, x_i) \ldots),$$

for $u = w \frown 0^{j+1}$. This can be done in time $|w \frown 0^j|^4$, if $w = r(K)$ (where K is the language determined by the code w) and $|x_1|, \ldots, |x_i| \leq |w \frown 0^j|^2$. Hence we have

$$(\exists^{lin} x_1)(\forall^{lin} x_2) \ldots (Q^{lin} x_i) R(w \frown 0^{j+1}, x_1, x_2, \ldots, x_i),$$

if and only if

$$(\exists x_1, |x_1| \leq |w \frown 0^j|^2)(\forall x_2, |x_2| \leq |w \frown 0^j|^2) \ldots (Q x_i, |x_i| \leq |w \frown 0^j|^2) \\ S(w \frown 0^j, x_1, x_2, \ldots, x_i).$$

This proves that condition (b) of Theorem 2.18 is satisfied. Thus L is a language which belongs to *PH* but not to *LinH*. □

We say that *LinH* (*PH* resp.) collapses if for some i, $LinH = \Sigma_i^{lin}$ ($PH = \Sigma_i^p$ resp.). This is equivalent to $\Sigma_i^{lin} = \Pi_i^{lin}$ ($\Sigma_i^p = \Pi_i^p$ resp.). We conjecture that neither *LinH* nor *PH* collapses and that *LinH* is a proper subclass of *PH*. But we have only the following result.

2.20 Corollary. Either $LinH \subsetneq PH$ or *LinH* does not collapse.

Proof. Suppose $LinH = \Sigma_i^{lin}$. Then $LinH \subsetneq PH$, since, by Theorem 2.19, $\Sigma_i^{lin} \subsetneq \Sigma_i^p \subseteq PH$. □

Let us define Σ_i^f, for f a function, to be the class of sets definable by formulae of the form

$$(\exists x_1, |x_1| \leq cf(|w|))(\forall x_2, |x_2| \leq cf(|w|)) \dots (Qx_i, |x_i| \leq cf(|w|)) \\ R(w, x_1, x_2, \dots, x_i),$$

where c is a constant, $R(w, x_1, x_2, \dots, x_i)$ is computable in time $cf(|w|)$ for every $x_1, \dots, x_i$, $|x_1| \leq cf(|w|), \dots, |x_i| \leq cf(|w|)$ and where Q is $\exists$ if i is odd and $\forall$ if i is even. Let

$$TH(f(n)) =_{df} \bigcup_i \Sigma_i^f \quad (\textit{Time Hierarchy } f).$$

The proof of Theorem 2.19 can be generalized to prove a statement similar to Theorem 2.17 for classes Σ_i^f and Σ_i^g where g grows faster then f for each fixed i. For classes $TH(f(n))$ this is an open problem (as we have mentioned above we do not know whether $LinH = PH$). Our last theorem shows that we can separate *PH* from slightly subexponential hierarchies. The bound $2^{\lfloor n^\varepsilon \rfloor}$ can be further decreased by iterating our method.

2.21 Theorem. For $\varepsilon > 0$, $PH \subsetneq TH(2^{\lfloor n^\varepsilon \rfloor})$.

First we shall prove some lemmas. Let L be a language and suppose that 0 is not in the alphabet of L; let f be a function such that $f(n) \geq n$ for all n. We define

$$L[f] =_{df} \{w \frown 0^{f(|w|)-|w|} \mid w \in L\}.$$

2.22 Lemma. Let f, g be time constructible functions.

(a) If $f(n) \geq g(n)$ for $n > n_0$, then for every $i \geq 0$, $\Sigma_i^g \subseteq \Sigma_i^f$.

(b) For every $i \geq 0$, $L \in \Sigma_i^f$ iff $L[f] \in \Sigma_i^{lin}$.

Proof. (a) is trivial, (b) follows easily from the definition of Σ_i^f and from the characterization of Σ_i^{lin} (Theorem 2.11 (a)). □

2.23 Lemma. Let f, g be time constructible functions; let g be growing faster than every polynomial. Suppose $TH(gf(n)) = PH$. Then $TH(ff(n)) \subseteq PH$.

Proof. Let the assumption of the lemma be satisfied. Let $L \in TH(ff(n))$. Using Lemma 2.22 (b) we get first $L[ff(n)] \in LinH$ and then $L[f] \in TH(f)$. Since $TH(gf(n)) = PH$, we get $L[f] \in PH$. Using Lemma 2.22 (b) again we have $L[pf(n)] \in LinH$ for some polynomial p. Hence $L \in TH(pf(n))$. Since g grows faster than any polynomial we have

$$TH(pf(n)) \subseteq TH(gf(n)) = PH\,.$$

Hence $L \in PH$. □

Proof of Theorem 22.1. Let $\varepsilon > 0$ be given. Let

$$f(n) = 2^{\lfloor n^{\varepsilon/2} \rfloor},$$
$$g(n) = 2^{(\lfloor \log_2 n \rfloor)^2}.$$

Suppose $PH = TH(2^{\lfloor n^{\varepsilon} \rfloor})$. Then we have

$$PH \subseteq TH(gf(n)) \subseteq TH(2^{\lfloor n^{\varepsilon} \rfloor}) = PH\,,$$

since it can be easily computed that $gf(n)$ grows faster than any polynomial but is smaller than $2^{\lfloor n^{\varepsilon} \rfloor}$. Thus $TH(gf(n)) = PH$. By Lemma 2.23 we have $TH(ff(n)) \subseteq PH$. We have the following inclusions:

$$PH \subseteq Exp \subsetneq Time\left(2^{2^{\lfloor n^{\varepsilon'} \rfloor}}\right) \subseteq TH\left(2^{2^{\lfloor n^{\varepsilon'} \rfloor}}\right) \subseteq TH(ff(n)) \subseteq PH\,,$$

where $\varepsilon/2 > \varepsilon' > 0$. The first inclusion is a natural simulation, the second one follows from Theorem 2.17 (ii), the third one is trivial and the fourth one follows from a simple computation and Lemma 2.22 (a). Thus we get a contradiction. □

*

We have defined several complexity classes, only a few of which will be mentioned in the following sections. The larger variety should help the reader to understand better the environment in which we are working, if we consider weak fragments. This short survey, however, cannot replace a textbook on such a broad subject.

3. Exponentiation, Coding Sequences and Formalization of Syntax in $I\Sigma_0$

(a) Introduction

In this chapter we shall present some basic constructions available in $I\Sigma_0$. First we want to show that it is not necessary to add the relation $x^y = z$ to $I\Sigma_0$, since it is definable there. To this end we have to develop the theory of sequences a little. Once we have the exponentiation relation we will be able to define a more efficient way of coding sequences. The second important task is to describe a formalization of syntax in $I\Sigma_0$. A formalization of syntax in $I\Sigma_1$ has been presented in Chap. I, Sect. 1(b). However $I\Sigma_0$ is a much weaker theory and it is not always possible to use classical techniques here. Usually formalization is very tedious work and the reader can find very few interesting ideas there. We cannot skip this part completely however as we have to persuade the reader that such a formalization is possible in a theory which at first looks very weak. Therefore we have decided on a compromise: instead of describing a particular formalization we develop some tools which can be easily applied to most standard situations. Clearly the reader of this book is not looking for formulae, but he is interested in ideas and techniques. In this section we shall consider only combinatorial problems related to the formalization of syntax. More about the formalization will be presented in Sect. 5.

The weaker the theory the more difficult it is to prove theorems. This applies to coding and formalization as well. Many concepts can be more easily handled in $I\Sigma_0 + \Omega_1$ than in $I\Sigma_0$. Since much research is going on in $I\Sigma_0 + \Omega_1$ (or equivalent theories) it is worthwhile mentioning these simpler formalizations.

Much of our intuition about natural numbers relies on exponentiation. Thus in order to persuade the reader about the possibility of a suitable formalization of (some of) the usual reasoning on numbers in $I\Sigma_0$, we prove the properties of exponentiation in great detail. After introducing the exponentiation we shall use more sketchy arguments.

Besides exponentiation we shall define one more important Σ_0-definable function: "the number of ones in the binary expansion". With these two concepts in hand, formalization of syntax will proceed quite smoothly.

The last subsection presents a general theorem which can be applied to formalize various syntactical concepts in $I\Sigma_0$.

In some estimates we shall use explicit numerals. The reason is that these numbers are easily computable and small.

(b) Sets and Sequences

Since $I\Sigma_0$ contains I_{open}, we know, by Chap. I, Sect. 1, that we can prove some elementary number theoretical facts. In order to be able to interpret other combinatorial objects, we need to formalize the concept of the sequence. Our task would not be so difficult, if we could start with the exponentiation *relation* in our language. But so far we do not know that this relation has a suitable definition in $I\Sigma_0$. An obvious requirement is that the defining formulae of the concepts be Σ_0, otherwise we cannot use $I\Sigma_0$. Therefore we cannot use the usual definition of the exponentiation relation in $I\Sigma_1$, which is not bounded.

Our approach is similar to that of [Nelson 86]. It is based on the observation that it is possible to talk about the digits in the binary expansion of a number, though (at first) we are not able to define that the digit in question is the i-th digit. The point is that we can define that a number is a power of 2 and then we can index the digits of a number by powers of 2. Namely, the i-th digit will have index 2^i. Then, clearly, we can easily talk about the relative position of digits. To code a sequence of numbers we shall use a pair of numbers. The first number will be the number determined by the concatenation of binary expansions of the numbers to be coded. The second one will be a binary code of the markers which determine beginnings and ends of the coded numbers. In fact we code sequences of arbitrary 0–1 words in such a way, but we shall use this possibility only after we define the exponentiation relation.

Let us consider an example. Suppose we want to code a sequence of 0–1 words

$$0011, \quad 101, \quad 010.$$

Then we take two numbers whose binary expansion is the following

$$11101010,$$
$$10001001001.$$

The first one is the concatenation of the words above (where we have to omit the first two 0's) and the second one is a sequence of markers which determines the partition. If this pair is considered to be a code of a sequence of numbers then it will code $(3, 5, 2)$.

Now we proceed formally in $I\Sigma_0$.

3.1 Definition (x *is a power of* 2).

$$Pow(x) \equiv (\forall p \leq x)(Prime(p) \,\&\, p \mid x \to p = \overline{2}) \,\&\, x \geq \overline{1}.$$

The relations *being a prime* and p *divides* x used in this definition are easily definable by Σ_0 formulae. The following is also easy to prove.

3.2 Lemma ($I\Sigma_0$).
(i) $Pow(\overline{1})$;
(ii) $x > \overline{1} \to (Pow(x) \equiv (\exists y)(x = \overline{2}y \,\&\, Pow(y)))$;
(iii) $Pow(x) \,\&\, Pow(y) \to Pow(xy) \,\&\, (x \mid y \vee y \mid x)$;
(iv) $Pow(x) \,\&\, Pow(y) \,\&\, (\overline{2}v + \overline{1})x = (\overline{2}u + \overline{1})y \to x = y$;
(v) $Pow(x) \,\&\, Pow(y) \,\&\, xz = y \to Pow(z)$;
(vi) $x > \overline{0} \to (\exists y)(x < y \leq \overline{2}x \,\&\, Pow(y))$.

The last lemma allows us to define:

3.3 Definition (y *is the least power of* 2 *larger than* x).

$$lpw(x) \equiv y = \min\{y; y \leq \overline{2}x + \overline{1} \,\&\, x < y \,\&\, Pow(y)\}\,.$$

3.4 Lemma ($I\Sigma_0$).
(i) $lpw(\overline{0}) = \overline{1}$;
(ii) $x > \overline{0} \to lpw(x) \leq \overline{2}x$. □

After we define the exponentiation and the length function $|x|$, it will be clear that $lpw(x) = 2^{|x|}$. Recall that we have a pairing function defined, say, by

$$(u, v) = 1/2((u + v)^2 + \overline{3}u + v)\,,$$

see I.1.18.

3.5 Definition (p *is a sequence*).

$$Seq(p) \equiv (\exists u, v \leq p)(p = (u, v) \,\&\, \neg\overline{2} \mid v \,\&\, lpw(u) \leq v)\,.$$

Here u is the sequence of 0–1 words of length at least 1 and v is the sequence of markers. At present we are not able to define the i-th element in the list, but we can easily define that a number occurs in it.

3.6 Definition.

$$\begin{aligned} x \in (u, v) \equiv (\exists y, y' \leq v)(\exists u_1, v_1 < y)(\exists u_2, v_2 \leq v)(&Pow(y) \,\&\, Pow(y') \\ &\,\&\, y' \geq \overline{2} \,\&\, x < y' \,\&\, u = u_2 y' y + xy + u_1 \\ &\,\&\, v = v_2 \overline{2} y' y + y' y + y + v_1)\,. \end{aligned}$$

Now we shall prove only the most basic properties of this coding, since we do not need more for defining and proving properties of exponentiation.

We show that the empty set has a code and that we can always add another element (number) in the sequence.

3.7 Lemma ($I\Sigma_0$).

(i) $(\forall x)(\text{not } x \in (\overline{0}, \overline{1}))$;

(ii) $(\forall p, z)(Seq(p) \rightarrow$
$\rightarrow (\exists q \leq \overline{9}p(z+\overline{1})^2)(Seq(q) \,\&\, (\forall x)(x \in q \equiv (x \in p \vee x = z))))$.

Proof. (i) is trivial. (ii) Let p be a sequence, $p = (u, v)$. Let $k = \max(\overline{2}, lpw(z))$. Define

$$u' = uk + z\,,$$
$$v' = vk + \overline{1}\,,$$
$$q = (u', v')\,.$$

Suppose $u > 0$. Then we have

$$u' \leq u * \overline{2}(z+\overline{1}) + z \leq \overline{3}(z+\overline{1})u\,,$$
$$v' \leq v * \overline{2}(z+\overline{1}) + \overline{1} \leq \overline{3}(z+\overline{1})v\,.$$

Since the pairing function is a quadratic polynomial, we have

$$q \leq \overline{9}(z+\overline{1})^2 p\,.$$

We leave to the reader to prove this bound for $u = 0$. Now we are going to prove $Seq(q)$. It is clear that v' is odd. To prove $lpw(u') \leq v'$, recall that we have $z < lpw(z) \leq k$ and $u < lpw(u)$ by definition and $lpw(u) \leq v$ by the assumption that $Seq(p)$. Thus

$$u' < uk + k \leq lpw(u)k \leq vk < v'\,.$$

Since $Pow(lpw(u)k)$, we have

$$lpw(u') \leq lpw(u)k \leq v'\,,$$

thus $Seq(p)$.

To show $z \in q$ put $y = 1$, $y' = k$, $v_1 = u_1 = 0$. The verification of the inequalities is straightforward then.

Now assume $x \in p$. We want to show $x \in q$. Let $y, y', u_1, v_1, u_2, v_2$ be witnesses of x being in p. Then

$$u' = u_2 y' yk + xyk + u_1 k + z\,,$$
$$v' = \overline{2}v_2 y' yk + y' yk + yk + v_1 k + \overline{1}\,.$$

Thus the witnesses for x being in q are respectively

$$yk, \quad y', \quad u_1 k + z, \quad v_1 k + \overline{1}, \quad u_2, \quad v_2\,.$$

We only have to check that

$$u_1 k + z < yk$$

and

$$v_1 k + \overline{1} < yk .$$

The inequalities are derived as follows

$$u_1 k + z < u_1 k + k = (u_1 + \overline{1})k \leq yk ,$$

and similarly the second one.

Now assume $x \in q$. We want to show $x \in p$ or $x = z$. Let us use $y, y', u_1, v_1, u_2, v_2$ for witnesses of x being in q. We consider two cases.

Case 1, $v_1 = 0$. Since v is odd and y' is even, y must be odd. As $Pow(y)$, we have $y = 1$. Thus

$$vk = v' - \overline{1} = \overline{2}v_2 y' + y' = (\overline{2}v_2 + \overline{1})y' ,$$

hence, since v is odd, $k = y'$. Then

$$uk + z = u' = u_2 k + x + u_1 ,$$

but $u_1 = 0$, since $u_1 < y = 1$, and $x < y' = k$, $z < k$. Thus $x = z$ by the uniqueness of the remainder (see I.1.15(4)).

Case 2, $v_1 > 0$. Then we have

$$vk = v' - \overline{1} = (\overline{2}v_2 y' + y' + \overline{1})y + v_1 - \overline{1} .$$

Since $v_1 - \overline{1} < y$, it is not possible that $y \mid k$. As y is a power of 2, we have $k \mid y$, (by Lemma 3.2(iii)). Hence $k \mid (v_1 - \overline{1})$. Thus we have

$$v = \overline{2}v_2 y' y k^{-1} + y' y k^{-1} + (v_1 - \overline{1})k^{-1} ,$$

which is not a formula in the language of arithmetic, but clearly can be expressed by such a formula. Similarly we have

$$uk = u_2 y' y + xy + u_1 - z .$$

Since $z < k$ and $k \mid y$, it must be $z \leq u_1$. Thus we can write

$$u = u_2 y' y k^{-1} + xy k^{-1} + (u_1 - z)k^{-1} .$$

All the inequalities for new witnesses follow immediately from the inequalities for the original witnesses. Thus $x \in p$. □

(c) The Exponentiation Relation

Our aim is to construct a Σ_0 formula $Exp(x,y,z)$ and to prove in $I\Sigma_0$ the following formulae:

(c.1) $$Exp(x,\overline{0},z) \equiv z = \overline{1};$$

(c.2) $$Exp(x,y+\overline{1},z) \equiv (\exists v)(Exp(x,y,v)\,\&\,z = vx).$$

Before starting the construction of the formula, let us observe that (c.1) and (c.2) imply some natural properties of Exp.

3.8 Lemma $(I\Sigma_0)$.
(i) $Exp(\overline{m},\overline{n},\overline{k}) \equiv \overline{m^n} = \overline{k}$;
(ii) $Exp(x,y,z)\,\&\,Exp(x,y,z') \to z = z'$;
(iii) $Exp(x,y,z)\,\&\,v \le y \to (\exists w \le z)Exp(x,v,w)$.

Proof. (i) – follows from (c.1),(c.2) and a similar fact for multiplication.

(ii) We would like to prove it by induction over y. We can use x as a parameter, but what to do with z and z'? The way out is to take the following formula instead of (ii):

$$(\forall z, z' < v)(Exp(x,y,z)\,\&\,Exp(x,y,z') \to z = z'),$$

and we can use v as a parameter of the induction.

(iii) is quite similar. □

The problem with the natural definition which uses recursion is that the size of the sequence that codes the course of values cannot be bounded by a polynomial, which is a necessary requirement for a Σ_0 formula. To overcome this difficulty we shall use sequences in which elements grow faster and hence they are shorter. The trick is to use the recursion on (binary) notation instead of the ordinary recursion.

Example. Let $y = 1101$ in binary; then we compute

$$x^0 = 1, \quad x^1 = x, \quad x^{11} = x^2{*}x, \quad x^{110} = (x^2{*}x)^2, \quad x^{1101} = ((x^2{*}x)^2)^2{*}x.$$

First we define an auxiliary formula (this is similar but not the same formula as in I.1.49).

3.9 Definition.

$$\begin{aligned} Exseq(x,s) \equiv Seq(s)\,\&\,(\overline{1},\overline{0}) \in s\,\&\,(\forall y,z \le s)[(z,y) \in s \to \\ \to (z = \overline{1}\,\&\,y = \overline{0}) \vee (y > \overline{0}\,\&\,(\exists v,w \le s)((y = \overline{2}v\,\&\,z = w^2 \\ \&\,(w,v) \in s) \vee (y = \overline{2}v + \overline{1}\,\&\,z = w^2x\,\&\,(w,v) \in s)))]. \end{aligned}$$

3.10 Lemma $(I\Sigma_0)$.

$$x \geq \overline{2} \,\&\, Exseq(x,s) \,\&\, (z,y) \in s \to (z,y) \leq \overline{4}z^2 .$$

Proof. Using induction on y we get $y \leq z$ and $z \geq 1$. Then an easy computation gives

$$(z,y) \leq (z,z) \leq \overline{4}z^2 . \qquad \square$$

3.11 Lemma $(I\Sigma_0)$.

$$(\forall x,s)(\forall z,y \leq s)(\exists t \leq \overline{30}z^{16})(x \geq \overline{2} \,\&\, Exseq(x,s) \,\&\, (z,y) \in s \to \\ \to Exseq(x,t) \,\&\, (z,y) \in t).$$

Proof. The sequence with a single element $(1,0)$ has number 30. Further we need the inequality

$$(*) \qquad z^{16} \geq \overline{9}(\overline{4}z^4 + 1)^2, \quad \text{for } z \geq \overline{2}.$$

We prove the lemma by induction over y with x and s as parameters. If $y = 0$, then $z = 1$, hence if the sequence t consists of a single element $(1,0)$, we have $t \leq 30z^{16}$. Now assume that $y' > 0$ and for every $z \leq s$ there exists a t satisfying the condition. We want to find t' for y' and some z' such that $(z',y') \in s$. By the definition of Exseq there exists $(z,y) \in s$ such that either

(i) $z' = z^2$ and $y' = \overline{2}y$,

or

(ii) $z' = z^2x$ and $y' = \overline{2}y + \overline{1}$.

By the induction assumption we have some t for (z,y). By Lemma 3.7 we can extend t to t' by adding (z',y') to it and so that

$$t' \leq \overline{9}t((z',y') + \overline{1})^2 .$$

Clearly t satisfies $Exseq(x,s)$. By lemma 3.10, we have

$$t' \leq \overline{9}t(\overline{4}(z')^2 + \overline{1})^2 .$$

If $z = 1$ then $y = 0$, hence $y' = 1$, $z' = x \geq 2$, and we have, by $(*)$,

$$t' \leq \overline{9} * \overline{30} * (\overline{4}(z')^4 + \overline{1})^2 \leq \overline{30}(z')^{16} .$$

If $z \geq 2$, then in case (i) we have, by $(*)$,

$$t' \leq \overline{9} * \overline{30}z^{16} * (\overline{4}z^4 + \overline{1})^2 \leq \overline{30}z^{32} = \overline{30}(z')^{16} ,$$

and in case (ii) we have

$$t' \leq \overline{9} * \overline{30}z^{16} * (\overline{4}z^4x^2 + \overline{1})^2 \leq \overline{30}z^{32}x^{16} = \overline{30}(z')^{16}. \qquad \square$$

Now we are ready to define the exponentiation.

3.12 Definition.

$$Exp(x,y,z) \equiv (z = \overline{1} \,\&\, y = \overline{0}) \vee (x \leq \overline{1} \,\&\, z = x \,\&\, y > \overline{0}) \\ \vee (\exists s \leq \overline{30}z^{16})(Exseq(x,s) \,\&\, (z,y) \in s).$$

We have written the formula so that it is bounded, however Lemma 3.11 permits us to drop the bound at the existential quantifier, which will be useful below. It is easy to check that the formula satisfies (c.1). To prove (c.2) we first derive the recurrent formulae which correspond to the recursion on notation and then derive (c.2) from them.

3.13 Lemma.

(c.3) $\quad Exp(x,y,z) \to Exp(x,\overline{2}y,z^2) \,\&\, Exp(x,\overline{2}y+\overline{1},z^2x)$;

(c.4) $\quad Exp(x,\overline{2}y,z) \to (\exists w \leq z)(w^2 = z \,\&\, Exp(x,y,w))$;

(c.5) $\quad Exp(x,\overline{2}y+\overline{1},z) \to (\exists w \leq z)(w^2x = z \,\&\, Exp(x,y,w))$.

Proof. (c.3) The lemma is trivial for $x \leq 1$ or $y = 0$. Suppose $x > 1$, $y > 0$ and $Exp(x,y,z)$. Let s be a witnessing sequence for $Exp(x,y,z)$ i.e. $Exseq(x,s)$ and $(z,y) \in s$. By Lemma 3.7, we can extend this sequence to s_1 (s_2 respectively) by adding another element $(z^2,2y)$ ($(z^2x,2y+1)$ respectively). Then we have $Exseq(x,s_i)$, for $i = 1,2$. Now we do not have to worry about the size of s_1 and s_2, since, by Lemma 3.12, we can replace s_1 and s_2 by suitably bounded ones.

(c.4) Again we can assume $x > 1$, $y > 0$. Let s be a witness for $Exp(x,\overline{2}y,z)$. Then by definition of $Exseq$ we have some w such that $z = w^2$ and $(w,y) \in s$. Again Lemma 3.12 takes care of the size of s. The proof of (c.5) is quite similar. $\square$

3.14 Lemma ($I\Sigma_0$).

$$Exp(x,y+\overline{1},z) \equiv (\exists v \leq z)(Exp(x,y,v) \,\&\, z = vx).$$

Observe that by Lemma 3.8 (ii) this is equivalent to (c.2).

Proof. Suppose that the formula is not true for some x, y and z. By $I\Sigma_0$, we can assume that it holds for every $y' < y$ and every $z' \leq z$. Consider two cases:

(i) *y is even*, say $y = 2p$. Then, by (c.4) and (c.3),

$$\begin{aligned} Exp(x, y + \overline{1}, z) &\equiv (\exists w \leq z)(w^2 x = z \,\&\, Exp(x, p, w)) \\ &\equiv (\exists w \leq z)(Exp(x, \overline{2}p, w^2) \,\&\, w^2 x = z)\,. \end{aligned}$$

Thus taking $v = w^2$ we get a contradiction.

(ii) *y is odd*, say $y = 2p + 1$. Then, similarly,

$$\begin{aligned} Exp(x, y + \overline{1}, z) &\equiv Exp(x, \overline{2}(p + \overline{1}), z) \\ &\equiv (\exists w \leq z)(w^2 = z \,\&\, Exp(x, p + \overline{1}, w))\,. \end{aligned}$$

Now by the induction assumption this is equivalent to

$$(\exists w, u \leq z)(Exp(x, p, u) \,\&\, ux = w \,\&\, w^2 = z)\,;$$

again by Lemma 3.7, we get

$$(\exists w, u \leq z)(Exp(x, \overline{2}p + \overline{1}, u^2 x) \,\&\, ux = w \,\&\, w^2 = z)\,;$$

The contradiction is obtained now by taking $v = u^2 x$. □

Now we can sum up what we have proved.

3.15 Theorem. The clauses (c.1) and (c.2) are provable in $I\Sigma_0$ for the Σ_0 formula of Definition 3.13. □

One can derive from (c.1) and (c.2) other natural properties of exponentiation in $I\Sigma_0$ such as, for instance,

$$\begin{aligned} &Exp(x, y_1, z_1) \,\&\, Exp(x, y_2, z_2) \to Exp(x, y_1 + y_2, z_1 z_2)\,; \\ &Exp(x, y_1, z) \,\&\, Exp(z, y_2, w) \to Exp(x, y_1 y_2, w)\,; \\ &Pow(z) \equiv (\exists y) Exp(\overline{2}, y, z)\,. \end{aligned}$$

Thus it is natural to ask whether the clauses (c.1) and (c.2) are sufficient for deriving all the other properties or we have to use sometimes also the explicit definition of the exponentiation relation. The answer is given by the following easy theorem.

3.16 Theorem. Let E be a new ternary relation symbol. Let $I\Sigma_0(E)$ be the theory $I\Sigma_0$ with the induction schema extended to bounded formulae containing E and with additional axioms

$$(c.1)' \qquad E(x, \overline{0}, z) \equiv z = \overline{1}\,;$$

$$(c.2)' \qquad E(x, y + \overline{1}, z) \equiv (\exists v)(E(x, y, v) \,\&\, z = vx)\,.$$

Then $I\Sigma_0(E)$ proves

$$E(x,y,z) \equiv Exp(x,y,z)\,.$$

Thus whatever we can prove about the exponentiation in $I\Sigma_0$, it can be derived from (c.1) and (c.2) (using the fact that it is defined by a bounded formula).

Proof. By induction over y we prove the following formula in $I\Sigma_0(E)$:

$$(\forall z \leq u)(E(x,y,z) \equiv Exp(x,y,z))\,. \qquad \square$$

Let us note that what we have described is not the only possible approach. In fact it is even possible to find a Σ_0 formula for exponentiation which does not use any concept of coding sequences, based on the representation of the power x^y in the form

$$x^y = (x-1)[(x-1)v + y] + 1\,,$$

for some v, see [Pudlák 83, A definition].

(d) Developing $I\Sigma_0 + \Omega_1$

For a moment we shall digress to $I\Sigma_0 + \Omega_1$. Recall that the theory $I\Sigma_0 + \Omega_1$ is the theory in the language L_0 with the following axioms

$$Q + I\Sigma_0 + \Omega_1$$

where Ω_1 is

$$(\forall x)(\exists y)(y = \omega_2(x))\,.$$

Let us note that often another function $\omega(x) = x^{|x|}$ is used instead of ω_2.

The function ω_2 can be expressed using the exponentiation and the length functions as follows.

$$\omega_2(x) = 2^{2^{2^{(||x|-1|-1)}}}$$

Thus formula $y = \omega_2(x)$ is just an abbreviation which uses the definition of exponentiation and the length function. Hence in order to able to state Ω_1 we must first construct a definition of exponentiation. If we want to work only in $I\Sigma_0 + \Omega_1$ and do not need the definition of exponentiation in $I\Sigma_0$, we can use a simpler definition of exponentiation.

3.17 Definition.

$$\begin{aligned} Exp_1(x,y,z) \equiv\ & (x = \overline{0}\,\&\, z = \overline{0}) \vee (x = \overline{1}\,\&\, z = \overline{1}) \\ & \vee (\exists s)[Seq(s)\,\&\,(z,y) \in s\,\&\,(\forall v \leq s)((v,\overline{0}) \in s \equiv v = \overline{1}) \\ & \&\,(\forall i < y)(\forall w < s)((w, i+\overline{1}) \in s \equiv (\exists v \leq s)((v,i) \in s\,\&\, w = vx))]\,. \end{aligned}$$

Here the defining formula is not bounded. One can show that s can be bounded by a polynomial in $\omega_2(z)$, and that there is no better bound. This has the following effect. If, for instance, we define the length function, which we need for the axiom Ω_1, by

$$|x|_1 = y \equiv (\exists z)(Exp_1(\overline{2}, y, z) \,\&\, x + \overline{1} \leq z < \overline{2}(x + \overline{1}))\,,$$

then already the statement

$$(\forall x)(\exists y)(y = |x|_1)$$

implies Ω_1. (Of course, this would be an awkward way to define $I\Sigma_0 + \Omega_1$.) The bound to the size of s in $Exp_1(x, y, z)$ can be proved in $I\Sigma_0 + \Omega_1$. Since induction for formulae with such a bound is also provable in $I\Sigma_0 + \Omega_1$ (Proposition 1.3), $Exp_1(x, y, z)$ is equivalent to $Exp(x, y, z)$, (and to any bounded definition satisfying the inductive clauses).

The simplified definition of exponentiation in $I\Sigma_0 + \Omega_1$ does not save us much work, since we still have to prove a lot of things. But for further concepts Ω_1 helps very much. Most of the syntactical concepts are naturally defined by recursion. The natural way to formalize definitions by recursion is to state that there exists a sequence (or a set) that codes the preceding values of the function. (This is called the course of values definition.) These sequences are usually of polynomial length, which is just captured by Ω_1. To get around with sequences of linear length requires additional tricks. We shall mention simpler definitions of the function *number of ones* and the property *being a term in* $I\Sigma_0 + \Omega_1$ in subsections (e) and (g).

(e) The Number of Ones in a Binary Expansion

We need another important Σ_0 definable function, the number of occurrences of 1 in the binary expansion of a number x. We shall denote this function by $nuon(x)$. As $nuon(x) \leq x$, the function is provably total in $I\Sigma_0$. Using complexity considerations, it is easy to show that *nuon* is Σ_0 definable, since it is computable in *LogSpace*. However, again, we need a Σ_0 definition which captures its properties in $I\Sigma_0$, thus we have to define it explicitly.

The basic properties of $nuon(x)$ are the following:

(e.1) $$nuon(\overline{0}) = \overline{0}\,;$$

(e.2) $$nuon(\overline{2}x) = nuon(x)\,;$$

(e.3) $$nuon(\overline{2}x + \overline{1}) = nuon(x) + \overline{1}\,.$$

Again, one can easily show in $I\Sigma_0$ that any two Σ_0 definitions satisfying the clauses above are equivalent.

Our next goal is to define $nuon(x)$ by a Σ_0 formula and prove (e.1–3) in $I\Sigma_0$. We shall proceed faster than in the previous section. From now on we shall use the usual notation for exponentiation instead of $Exp(x,y,z)$. Thus if x^y is used in a formula without any comment, it means that the existence of this number is somehow guaranteed.

3.18 Definition.

(i) $sgm(x,i,j) = y \equiv (\exists x_1, x_2 \leq x)(x = x_1 2^j + y2^i + x_2 \,\&\, x_2 < 2^i \,\&\, y < 2^{j-i})$ for $i < j$,
otherwise 0;

(ii) $bit(x,i) = sgm(x,i,i+\overline{1})$;

(iii) $|x| = \min\{y \mid x < 2^y\}$.

Here $sgm(x,i,j)$ is the part of x between the i-th and $(j-1)$-th digit; $bit(x,i)$ is the i-th dyadic digit of x (see I.1.31); $|x|$ is the length of the binary representation of x. (The length function has been defined before, formula (iii) gives an explicit definition of it in terms of the exponentiation.) We leave to the reader to check that formula (ii) is equivalent to a Σ_0 formula and that $2^{|x|}$ has the same meaning as $lpw(x)$.

3.19 Definition.

$$Nuonseq(s,x,y) \equiv Seq(s)\,\&\,(\forall z \leq s)((z,\overline{0}) \in s \equiv z = \overline{0})\,\&\,(y,|x|) \in s\,\&$$
$$\&\,(\forall i < |x|)(\forall z \leq s)((z,i+\overline{1}) \in s \equiv (\exists v \leq s)((v,i) \in s$$
$$\&\, z = v + bit(x,i)))\,.$$

3.20 Lemma. There exists an absolute constant K such that $I\Sigma_0$ proves

$$2^{|x|^2}\text{exists} \to (\exists s \leq 2^{K*|x|^2})(\exists y \leq x)\ Nuonseq(s,x,y)\,.$$

(The antecedent should be written more precisely as

$$(\exists z)Exp(\overline{2},|x|^2,z)\,.$$

Proof. The formal proof in $I\Sigma_0$ would be similar to the proof of Lemma 3.11. Here we only check the size of s in the standard model. The sequence s contains $|x|$ elements of the form (z,i) where $z \leq i \leq |x|$. Thus its length is at most

$$K * |x| * \|x\| \leq K * |x|^2\,.$$ □

3.21 Lemma $(I\Sigma_0)$.

$$Nuonseq(s,x,y)\,\&\,Nuonseq(t_1,\overline{2}x,y_1) \to y_1 = y\,;$$
$$Nuonseq(s,x,y)\,\&\,Nuonseq(t_2,\overline{2}x+\overline{1},y_2) \to y_2 = y+\overline{1}\,.$$

Proof. Using induction over $i = 0, \ldots, |x|$ show that

$$(z, i + \overline{1}) \in t_1 \equiv (z, i) \in s\,;$$
$$(z, i + \overline{1}) \in t_2 \equiv (z, i + \overline{1}) \in s\,. \qquad \square$$

In $I\Sigma_0 + \Omega_1$ we can define $nuon(x) = y$ by

$$nuon_1(x) = y \equiv (\exists s) Nuonseq(s, x, y)\,,$$

since by Lemma 3.20 we know that s can be bounded by a function which is total in $I\Sigma_0 + \Omega_1$, (the subscript is used to distinguish different formulae). Then we can use Lemma 3.21 to prove clauses (e.1–e.3). Unfortunately we cannot use this formula in $I\Sigma_0$ as there is no polynomial bound to such an s. We have to use a more complicated construction. The idea is similar to that of Nepomnjaščij presented in Sect. 1 of this chapter. Namely, we cut x into segments x_i such that on each segment x_i we can use $nuon_1(x_i)$, because the witnessing sequences are bounded by a polynomial in in x (i.e. the pieces are small enough). Then $nuon(x)$ is computed as the sum of the values $nuon_1(x_i)$.

We need the following function, whose graph is Σ_0 definable in $I\Sigma_0$:

$$lf(x) = 0 \qquad \text{if } x = 0\,,$$
$$= |x^x| \quad \text{otherwise}.$$

3.22 Definition. Let $nuon(0) = 0$ and, for $x > 0$, let

$$\begin{aligned} nuon(x) = y \equiv (\exists t)\big[&Seq(t)\,\&\,(\forall z \le t)[(z, \overline{0}) \in t \equiv z = \overline{0}] \\ &\&\,(\forall i \le x)(lf(i) < |x| \le lf(i + \overline{1}) \to (y, i + \overline{1}) \in t) \\ &\&\,(\forall i \le x)(\forall z \le t)[lf(i) < |x| \to ((z, i + \overline{1}) \in t \\ &\quad \equiv (\exists v \le t)((v, i) \in t\,\& \\ &\qquad \&\, z = v + nuon_1(sgm(x, lf(i), lf(i + 1)))))]\big]\,. \end{aligned}$$

In order to show that this formula is Σ_0 over $I\Sigma_0$ we have to find a polynomial bound for t and for the sequences s in $nuon_1$. First observe that, for a sufficiently large constant c,

$$i \ge c\frac{|x|}{\|x\|} \to |i^i| \ge |x|\,.$$

Thus we need only those i's which are smaller than this bound. Now we have to estimate

$$sgm(x, lf(i), lf(i + 1))\,.$$

The length of this number is bounded by $lf(i+1)-lf(i)$. Using the inequality $a < 2^{|a|} \leq 2a$, for $a > 0$, we get, for $i > 0$,

$$2^{|(i+1)^{i+1}|-|i^i|} = \frac{2^{|(i+1)^{i+1}|}}{2^{|i^i|}} < \frac{2(i+1)^{i+1}}{i^i} = 2(i+1)(1+\frac{1}{i})^i,$$

which tends to $2e(i+1)$ as $i \to \infty$. Thus we have, for some constants c', c'' and $i > 0$,

$$\text{(e.4)} \qquad sgm(x, lf(i), lf(i+1)) \leq 2^{lf(i+1)-lf(i)} = 2^{|(i+1)^{i+1}|-|i^i|} \leq c'(i+1) \leq c'' \frac{|x|}{\|x\|}.$$

Now, by Lemma 3.20, $nuon_1(sgm(x, lf(i), lf(i+\overline{1})))$ is witnessed by a sequence s such that

$$s \leq 2^{K|sgm(x,lf(i),lf(i+1))|^2} \leq 2^{K|c''\frac{|x|}{\|x\|}|^2} \leq 2^{K'|x|},$$

for some constant K', which is a bound polynomial in x. The sequence t consists of $c\frac{|x|}{\|x\|}$ elements. The elements have form (z, i) with

$$i \leq c\frac{|x|}{\|x\|}, \quad z \leq |x|,$$

thus the length of (z, i) is linear in $\|x\|$. Hence the length of t is linear in $|x|$, i.e. t is bounded by a polynomial in x.

The proof above has been done in the standard model. The proof in $I\Sigma_0$ would be more complicated, but very similar to the proof of Lemma 3.11. We sketch briefly this argument.

First we can estimate in $I\Sigma_0$

$$(1+1/i)^k \leq 1 + 2k/i + 2k^2/i^2,$$

for $k \leq i$, hence we get $(1+1/i)^i \leq 5$ and we can use (e.4). Secondly we need to estimate elements (z, i) of the sequence t for $|i^i| \leq |x|$. This is done by showing that $nuon_1(a) \leq |a|$, whenever defined (by induction on the length of a), from which we get $(z, i) \leq p(|x|)$ for some polynomial p. For a given x take the largest j such that $|j^j| \leq |x|$. By induction over j we show that the sequence t can be bounded by $q(i^i)$ for a suitable polynomial which we shall specify later. Suppose that such a bound holds for j and let x be such that $j+1$ is the maximal such that $|(j+1)^{j+1}| \leq |x|$. Let $x' = sgm(x, \overline{0}, j^j)$. Thus we have some t' for x' with $t' \leq q(j^j)$. To obtain t for x, we only need to add

an element $(z, j+1)$ to t. Since $(z, j+1) \leq p(|x'|)$ and $j+1$ is about $\frac{|x'|}{\|x'\|}$, we have, for a suitable polynomial r, $(z, j+1) \leq r(j)$. Thus, by Lemma 3.7

$$t \leq t' * 9(r(j)+1)^2 \leq q(j^j) * r'(j)\,.$$

The induction can now be completed, if we ensure that

$$q(j^j) * r'(j) \leq q((j+1)^{j+1})\,,$$

which can be done by taking polynomial q sufficiently large with respect to r'.

Thus we have shown:

3.23 Lemma. In $I\Sigma_0$ the formula defining $nuon(x) = y$ is equivalent to a Σ_0 formula. □

Remark. There are, of course, other possible ways to divide the binary expansion of x for the definition of *nuon*. The reasons why we have chosen the partition determined by the function $lf(i)$ are the following:

(1) it does not depend on x, hence we can easily use induction;
(2) it grows smoothly, (while e.g. $i * |i|$ makes big jumps).

3.24 Lemma.

(i) The formula $nuon(x) = y$ defines a function in $I\Sigma_0$.
(ii) The formula $nuon(x) = y$ satisfies (e.1–e.3) provably in $I\Sigma_0$.

Proof. The proof of (i) is omitted. The proof of (e.1) follows from the definition. To prove (e.2) and (e.3) we would like to simulate the proof of Lemma 3.21. Now we cannot code the sequence of the values of $nuon(x')$ for all initial segments x' of x by a number, but we can define these values by a Σ_0 formula. Namely, let t be a witness of $nuon(x) = y$, let $k \leq |x|+1$. Then the value $z = nuon(x')$ for the initial segment x' of x of length k is defined by

$$\varphi(z,k,t,x) \equiv (\exists i, v)(|i^i| < k \leq |(i+1)^{i+1}| \,\&\, (v,i) \in t \\ \&\, z = v + nuon_1(sgm(x, |i^i|, k)))\,.$$

Similarly as above one can show that φ is Σ_0 in $I\Sigma_0$. Now let t' be a witness for $nuon(2x) = y'$. Since, by Lemma 3.21, $nuon_1(2a) = nuon_1(a)$, we have, by induction on k,

$$\varphi(z,k,t,x) \equiv \varphi(z,k,t',2x)\,.$$

But we have also

$$\varphi(z, |x|+1, t, x) \equiv z = y\,,$$
$$\varphi(z, |2x|, t', 2x) \equiv z = y'\,,$$

hence $y = y'$. The proof of (e.3) is similar. □

3.25 Lemma.

$$nuon(x2^{|y|} + y) = nuon(x) + nuon(y)\,.$$

Proof. First we show that

$$x2^{|2y|} + 2y = 2(x2^{|y|} + y)\,,$$
$$x2^{|2y|} + 2y + 1 = 2(x2^{|y|} + y) + 1\,,$$

and then use (e.1–e.3) and induction *PIND* (which is derivable from the ordinary induction, see Sect. 4) over y. □

Also here there is an alternative way to define the concept *nuon*. This was suggested by S. Buss, (unpublished). The idea is to count ones in the straightforward way as in $nuon_1$ except that we include in the sequence of *Nuonseq* only the bits which have been changed. Thus e.g. the following sequence

$$1, 10, 10, 11, 100, 101, 101, 110, 111, 1000\,,$$

will be replaced by

$$1, 10, \quad , 1, 100, 1, \quad , 10, 1, 1000\,.$$

Then we need at most $2n$ bits to count up to the number n (plus bits needed to separate the members of the sequence).

(f) Coding Sequences

Using the function *nuon* we are now able to define a very efficient coding, which is close to the information theoretical lower bound. Before defining *nuon* we were able to say that an element x belongs to a sequence and that an element x precedes an element y in sequence s. Now we can define that x is the i-th element of s.

3.26 Definition.

$$\begin{aligned}(s)_i = x \text{ if } &(\exists u, v \le s)(\exists y, y' \le v)(\exists u_1, v_1 < y)(\exists u_2, v_2 \le v)(Seq(s)\\ &\& s = (u, v) \,\&\, Pow(y) \,\&\, Pow(y') \,\&\, y' \ge \overline{2} \,\&\, x < y'\\ &\& u = u_2 y' y + xy + u_1 \,\&\, v = v_2 \overline{2} y' y + y' y + y + v_1\\ &\& i = nuon(v_1))\,.\\ = \overline{0} &\text{ otherwise.}\end{aligned}$$

Thus $(s)_i = x$ is defined by the same formula as $x \in s$ (Definition 3.6) except that we have added the condition $i = nuon(v_1)$. The meaning of this condition

is clear: i is the number of markers before the beginning of the occurrence of x in u. Next we define the length of a sequence and concatenation of two sequences.

3.27 Definition.

(i) $lh(s) = nuon(v) - 1$, if $Seq(s) \,\&\, (\exists u \le s) s = (u, v)$
$= 0$ otherwise.

(ii) For s, t such that $Seq(s)$, $Seq(t)$, $s = (u, v)$ and $t = (u', v')$

$$s \frown t = (u'', v''),$$

where

$$u'' = u' 2^{|v|-1} + u,$$
$$v'' = (v' - 1) 2^{|v|-1} + v;$$

otherwise we set $s \frown t = \overline{0}$.

Note that these functions are defined by Σ_0 formulae. We shall state the most important properties of concatenation. The role of the empty sequence is played by $(1, 0)$.

3.28 Lemma ($I\Sigma_0$). For all x, y, z such that $Seq(x)$, $Seq(y)$ and $Seq(z)$

(i) $(\overline{1}, \overline{0}) \frown x = x \frown (\overline{1}, \overline{0}) = x$;
(ii) $x \frown y = x \frown z \to y = z$;
(iii) $y \frown x = z \frown x \to y = z$;
(iv) $x \frown (y \frown z) = (x \frown y) \frown z$;
(v) $lh(\overline{1}, \overline{0}) = \overline{0}$, $lh(x \frown y) = lh(x) + lh(y)$.

Proof. The proofs are just tedious verifications. Therefore we prove only (ii) to illustrate the proof technique.

Let $x = (u(x), v(x))$, $y = (u(y), v(y))$, $z = (u(z), v(z))$. Suppose $x \frown y = x \frown z$, then

$$(v(x) - 1) 2^{|v(y)|-1} + v(y) = (v(x) - 1) 2^{|v(z)|-1} + v(z).$$

Since the function $a \mapsto 2^{|a|-1}$ is nondecreasing, we have $v(y) = v(z)$. Further we have

$$u(x) 2^{|v(y)|-1} + u(y) = u(x) 2^{|v(z)|-1} + u(z).$$

Since, by the definition of Seq, $|u(y)| < |v(y)|$, $|u(z)| < |v(z)|$ and since $v(y) = v(z)$, we have $u(y) = u(z)$ by the lemma about division with reminder Theorem I.1.10 (6). □

Since $x, y \mapsto x \frown y$ is a total function in $I\Sigma_0$, it must be bounded by a polynomial in x, y. We shall state an explicit bound now. The proof is just an easy computation.

3.29 Proposition ($I\Sigma_0$). $s \frown t \leq \overline{64}st$.

The maximal element of a sequence s is trivially definable by a Σ_0 formula, it will be denoted by $\max(s)$. We shall use $\max(s)$ and $lh(s)$ to write down an explicit bound to the size of t which codes the same sequence of numbers as s. We cannot bound s itself, since numbers in s can be represented by 0-1 sequences with many 0's before the first 1. This is a drawback of our definition.

3.30 Proposition.

$$Seq(s) \to (\exists t)(Seq(t) \,\&\, lh(t) = lh(s) \,\&\, (\forall 0 \leq i < lh(s))((t)_i = (s)_i) \\ \,\&\, |t| \leq (2|\max(s)| + 4) * lh(s) + 2).$$

Proof. By induction on j, $j = lh(s), lh(s) - 1, \ldots, 0$, we shall prove the following formula

$$(\exists t)(Seq(t) \,\&\, lh(t) = lh(s) - j \,\&\, (\forall 0 \leq i < lh(s) - j)((t)_i = (s)_{j+i}) \\ \,\&\, |t| \leq (2|\max(s)| + 4) * (lh(s) - j) + 2).$$

For $j = lh(s)$ put $t = (0,1) = 1$. Suppose it is true for $0 < j \leq lh(s)$, i.e. we have some t' for j. Then let t be t' extended by $(s)_{j-1}$ as described in the proof of Lemma 3.7. It is not difficult to check that $(t)_i = (t')_{i-1}$ for $i = 1, \ldots, lh(s) - j$, and $(t)_0 = (s)_{j-1}$. Hence $(t)_i = (s)_{j-1+i}$. Further we have

$$t \leq 9t'((s)_{j-1} + 1)^2 \leq 9t'(\max(s) + 1)^2.$$

An easy computation gives us

$$|t| \leq 4 + |t'| + 2|\max(s)| \leq (2|\max(s)| + 4)(lh(s) - j + 1) + 2.$$

Thus the formula is proved. Taking $j = 0$ we obtain our proposition. □

Suppose a finite sequence of numbers is defined by a Σ_0 formula. In $I\Sigma_0 + Exp$ such a sequence always has a code. When does it have a code in $I\Sigma_0$, i.e. which comprehension principles hold in $I\Sigma_0$? The answer is that there is only one essential restriction: the length of the code must be linear in the parameters. We state an example of a replacement schema which is valid in $I\Sigma_0$.

3.31 Proposition. For every Σ_0 formula $\varphi(x,y)$ $I\Sigma_0$ proves

$$(\forall x \leq u)(\exists y \leq v)\varphi(x,y) \,\&\, (\exists z)(z = (v+2)^u) \to \\ \to (\exists s)(lh(s) = u \,\&\, (\forall x < u)(\varphi(x,(s)_x) \,\&\, (s)_x \leq v)).$$

Proof. By induction on u; similar to the proof of Proposition 3.30. □

(g) Syntactical Concepts

We shall use natural representation of syntactical objects as particular sequences (words) in a finite alphabet. Since we have concatenation, we do not have problems with the properties which can be derived form definitions based on rudimentary formulae. Recall that x *is a part of* y is defined by

$$x \subseteq_p y \equiv (\exists u, v \leq y)(y = u \frown x \frown v),$$

which is a Σ_0 formula. (Rudimentary formulae are the formulae in the language $\{=, \frown\}$ with bounded quantifiers of the form $\exists x \subseteq_p y$ and $\forall x \subseteq_p y$.) Thus, for instance, once we have the concept of a formula, we can define the concept of being a subformula and prove its properties. Similarly there are no problems with defining that a variable does (not) occur in a formula etc.

What is not quite obvious is that we can define the concept of terms and formulae by Σ_0 formulae. Actually, it is sufficient to define only terms, since terms are essentially trees and other syntactical concepts can be represented as particular labelled trees with some additional properties (these properties usually defined by rudimentary formulae).

We shall consider a simplified situation: We have four symbols, a variable **v**, brackets **(**,**)**, and a binary operation symbol **o**. Then the terms are the smallest set satisfying:

(1) **v** is a term;
(2) if s and t are terms, then $(s \,\mathbf{o}\, t)$ is a term.

A formula $Term(x)$ defining the concept of terms must satisfy the following clauses:

(g.1) $Term(\mathbf{v})$;

(g.2) $Term(x) \,\&\, Term(y) \to Term(\mathbf{(} \frown x \frown \mathbf{o} \frown y \frown \mathbf{)})$;

(g.3) $Term(x) \to x = \mathbf{v} \vee (\exists p, q)(Term(p) \,\&\, Term(q) \,\&\, x = \mathbf{(} \frown p \frown \mathbf{o} \frown q \frown \mathbf{)})$.

The first two express formally (1) and (2), (g.3) corresponds to the clause that "*Term* is the smallest set satisfying (1) and (2)". In fact we can prove that any two Σ_0 formulae satisfying (g.1–3) are equivalent in $I\Sigma_0$. It is exactly the same situation as with exponentiation and function *nuon* of subsection (e).

In $I\Sigma_0 + \Omega_1$ we can use the following simple definition:

$$Term_1(x) \equiv (\exists s)(Seq(s) \,\&\, x \in s \,\&\, (\forall y \in s)(y = \mathbf{v} \vee (\exists p, q)(p \in s \,\&\, q \in s \,\&\, y = \mathbf{(} \frown p \frown \mathbf{o} \frown q \frown \mathbf{)}))).$$

From now on we agree to omit the sign for concatenation whenever there is no danger of confusion. The proof of (g.1–3) for $Term_1$ in $I\Sigma_0 + \Omega_1$ is easy.

Our approach in $I\Sigma_0$ will be based on counting brackets. The occurrences of brackets (or other symbols) can be easily coded by occurrences of ones in the binary expansion of a number. Thus we can again use function ***nuon***. Since this transformation is easy, we shall use less formal descriptions of formulae.

3.32 Definition. (well-bracketing, sub-well-bracketing, term)

(i) $WB(x) \equiv$ (1) the number of "(" in x equals to the number of ")" in x;
(2) the number of ")" in each proper initial segment of x is less than the number of "(" in it;
(3) x is nonempty.

(ii) $SubWB(x,y) \equiv WB(x) \,\&\, WB(y) \,\&\, x \subseteq_p y$.

(iii) $Term(x) \equiv WB(x) \,\&\, (\forall y)(SubWB(y,x) \to y = \mathbf{v}$
$\vee\, (\exists p,q)(SubWB(p,x) \,\&\, SubWB(q,x) \,\&\, y = (p \circ q)))$.

3.33 Lemma. $WB(x)$, $SubWB(x,y)$ and $Term(x)$ are Σ_0 over $I\Sigma_0$.

Proof. These formulas use Σ_0 defined predicates and functions (for counting brackets use the function ***nuon***); the quantification can be made bounded. □

3.34 Lemma. $I\Sigma_0$ proves (g.1–3) for the definition of $Term(x)$ above.

Proof. (g.1) is obvious. To prove (g.2), assume $Term(x)$ and $Term(y)$ and let $z = (x \circ y)$. The condition $WB(z)$ is easy. Let u be a sub-well-bracketing of z. Clearly it is now sufficient to show that u is either a part of x or a part of y or u equals to z. If u is not a part of x nor a part of y, it must contain the first occurrence of "(" or the occurrence of $\circ$ between x and y or the last occurrence of ")".

(1) Suppose u contains the first occurrence of "(", then it must contain also the last occurrence of ")", otherwise condition (1) of $WB(u)$ would be violated.

(2) The case with the last occurrence of ")" is symmetric.

(3) Suppose u contains $\circ$ between x and y. Then u is not $\mathbf{v}$, hence it contains an end segment of x or an initial segment of y. In the first case it must contain the first occurrence of "(" and in the second case it must contain the last occurrence of ")", since otherwise condition (2) of $WB(u)$ would be violated. Hence again $u = z$ and condition (g.2) is proved.

Now we prove (g.3). Suppose $Term(x)$. If $x = \mathbf{v}$, then there is nothing to prove. If not, then $x = (p \circ q)$ for some p, q such that $SubWB(p,x)$ and $SubWB(q,x)$. To prove that p and q are terms, we have to check that each sub-well-bracketing y of p or of q has the required form. But each such y is also a sub-well-bracketing of x, hence by definition it must have this form.

□

Note that we can think of *WB* as a definition of arbitrary finite trees and Term as a definition of finite binary trees. The last property of terms that we prove is sometimes called *unique readability*.

3.35 Lemma ($I\Sigma_0$).

$$Term(x)\,\&\,Term(y)\,\&\,Term(p)\,\&\,Term(q)\,\&\,(x \circ y) = (p \circ q) \rightarrow \\ \rightarrow x = p \,\&\, y = q\,.$$

Proof. Assume the antecedent is true. Then p and q are sub-well-bracketings of $(x \circ y)$. It follows now from the proof of Lemma 3.34 that neither p nor q can contain the occurrence of $\circ$ between x and y. Hence the only possibility is that $x = p$ and $y = q$. □

Now we consider *substitution*. We have defined terms in a very restricted language; the following proposition holds for terms in the usual general sense.

3.36 Proposition. The relations *the term u is the result of substituting s for all (some,* respectively*) occurrences of the variable* $\mathbf{v}$ *in* t are Σ_0 definable.

Proof. We need to express that t is of the form

$$p_1 \frown \mathbf{v} \frown p_2 \frown \mathbf{v} \frown \ldots \frown p_{n-1} \frown \mathbf{v} \frown p_n\,,$$

where $\mathbf{v}$ does not occur (may occur, respectively) in $p_1, p_2, \ldots, p_n$, and u is of the form

$$p_1 \frown s \frown p_2 \frown s \frown \ldots \frown p_n \frown s \frown p_n\,.$$

This can be done by introducing two 0-1 sequences w_1, w_2 such that w_1 marks occurrences of $\mathbf{v}$ in t and w_2 marks occurrences of s in u (not necessarily all). Then we state that the word between the i-th and $i+1$-st marked occurrence of $\mathbf{v}$ in t is equal to the word between the i-th and $i+1$-st marked occurrence of s in u. To express that an occurrence is the i-th one we use, of course, our function *nuon*. □

Since the definition of substitution is quite straightforward, the usual properties are easily provable in $I\Sigma_0$ and we shall not present them here. In $I\Sigma_0 + \Omega_1$ we can define the total ternary operation

$$t, \mathbf{v}, s \mapsto t(\mathbf{v}/s)\,,$$

where $t(\mathbf{v}/s)$ stands for the result of substituting s for all occurrences of variable $\mathbf{v}$. This is because we can bound the length of $t(\mathbf{v}/s)$:

$$|t(\mathbf{v}/s)| \leq \text{const} * |t| * |s|\,.$$

Since the bound is, essentially, the best possible and it is not linear, this function is not total in $I\Sigma_0$. This causes some problems, when we develop metamathematics in $I\Sigma_0$. For example, we cannot replace occurrences of a variable by a term in a proof.

The purpose of this section is to study combinatorial concepts needed for formalization. It should be clear by now that these concepts are quite simple and can be handled in $I\Sigma_0$, unless they are functions growing more than polynomially. In the next section we prove a general theorem which covers some standard cases.

Let us consider the concept of a proof as an example. Suppose we formalize proofs as sequences of formulae. Then we need to write down that each element of the sequence is a formula which follows from the preceding ones by a rule. The relation that a rule defines on formulae, is usually constructed from notions whose definability and properties have been shown in $I\Sigma_0$ (concatenation, substitution, occurrence etc.). On the other hand the provability predicate which we obtain in this way does not have the basic closure properties in $I\Sigma_0$. This is because it is a Σ_1 concept.

(h) Formalizations Based on Context-Free Grammars

After working for some time in a formal system one gets a feeling, usually quite a clear one, of what can be formalized. Repeating the same tricks in formalizations again and again becomes boring. Then comes a dream: to prove a general theorem which would take care of all situations. Usually it remains only a dream, because if we want to state a comprehensible and memorable theorem, we have to give up a lot of its possible applications. We are going to state and prove a theorem which is such a compromise.

We shall use the concept of a context-free grammar. A *grammar* is determined by two alphabets and rewriting rules. A *rewriting rule* is a pair of words (u, v). By an application of the rule (u, v) to a word w we mean the replacement of an occurrence of u in w by v. The rules can be applied in arbitrary order to arbitrary occurrences of subwords. One of the alphabets consists of so called *nonterminal symbols* the other consists of *terminal symbols*. We are interested in the words which consist of terminal symbols only and which can be derived from a specified initial nonterminal symbol using rewriting rules. This is *the language* determined by the grammar. A *context-free grammar* is a grammar in which the first word of the rewriting rule is always a single symbol. In context-free grammars we can interpret nonterminal symbols as *concepts* (e.g. "noun", "verb" in natural languages, or "term", "procedure" in programming languages). The rules can be thought of as a kind of inductive clauses, which determine the structure of the words in the language. We know that it is important to have proofs of inductive clauses

in $I\Sigma_0$. This is exactly what our theorem will do. Examples will be shown at the end of the section.

Let us state a formal definition of context-free grammars and introduce the standard notation. A *context-free grammar* is a quadruple $G = (N, T, P, S_0)$, where N and T are finite sets, $N \cap T = \emptyset$, $S_0 \in N$, $P \subseteq N \times (N \cup T)^+$, where $^+$ denotes nonempty words in a given alphabet. (The standard definition also allows the derivation of empty words, but it would cause us some technical complications, therefore we consider only languages without empty words; the results that we shall prove do not depend on this restriction.) The elements of N (T respectively) are called *nonterminal symbols* (*terminal symbols*, respectively), S_0 is the *initial nonterminal symbol*, P is a set of *rewriting rules*. Let $u, w \in (N \cup T)^+$; we shall write

$$u \Rightarrow w\,,$$

(or $u \underset{G}{\Rightarrow} w$ to indicate which grammar we are using), if there are $A \in N$ and $v \in (N \cup T)^+$ such that A occurs in u, $(A, v) \in P$ and w results from u by replacing an occurrence of A by v. The relation $\overset{*}{\Rightarrow}$ denotes the transitive closure of $\Rightarrow$. Thus $u \overset{*}{\Rightarrow} w$ means that $u = w$ or u can be rewritten to w by applications of the rules of the grammar. Note that the rewriting process is nondeterministic: we can choose rules and occurrences of letters. The set of all words $w \in T^+$ such that $S_0 \overset{*}{\Rightarrow} w$ is *the language generated by* G. The languages generated by context-free grammars are called *context-free languages*.

We shall add another unessential *restriction* that there is no rule of the form $(A, B) \in P$ with $A, B \in N$. In order to simplify notation, we shall omit the sign for concatenation in the rest of this section.

Our aim is to formalize the property of being derivable from A; this is the meaning of $\varphi_A(w)$ below.

3.37 Theorem. Let G be a context-free grammar. Then one can construct a family of Σ_0 formulae φ_A for $A \in N$ with the following property: If

$$A \Rightarrow w_{i1} B_{i1} w_{i2} B_{i2} \dots B_{in_i - 1} w_{in_i}\,, \quad i = 1, \dots, k\,,$$

are all the rules of G with A on the left side and $w_{i1}, w_{i2}, \dots, w_{in_i} \in T^*$, $B_{i1}, B_{i2}, \dots, B_{in_i - 1} \in N$, then $I\Sigma_0$ proves

$$\text{(g.1)}\quad \varphi_A(x) \equiv \bigvee_{i=1}^{k} (\exists y_1) \dots (\exists y_{n_i - 1})(\varphi_{B_{i1}}(y) \,\&\, \dots \,\&\, \varphi_{B_{in_i - 1}}(y) \,\&\, x = w_{i1} y_1 w_{i2} y_2 \dots y_{n_i - 1} w_{in_i})\,,$$

(where we admit $n_i = 1$).

Proof. Let a context-free grammar $G = (N, T, P, S_0)$ be given. We consider an auxiliary grammar $\underline{G} = (N, \underline{T}, \underline{P}, S_0)$ defined by adding a new terminal $\underline{A}$ for each nonterminal $A \in N$ of G and two new terminal symbols [,], and by replacing each rule of G

$$A \Rightarrow w$$

by

$$A \Rightarrow \underline{A}[w]\,,$$

($\underline{A}[w]$ is the concatenation of $\underline{A}$, [, w and].) The idea behind this is that $\underline{G}$ codes the syntactical trees of derivations in G.

The words $w \in (N \cup \underline{T})^+$ which are derivable from some A in $\underline{G}$ can be defined by the following clauses:

(1) w is a well-bracketing with respect to [and];
(2) w is of the form $\underline{A}[u]$ for some $A \in N$;
(3) each occurrence of [is preceded by some $\underline{A}$, for $A \in N$;
(4) if

(g.2) $$\underline{A}[w_1\underline{B}_1[v_1]w_2 \dots w_{n-1}\underline{B}_{n-1}[v_{n-1}]w_n]$$

is a sub-well-bracketing of w, then

(g.3) $$A \underset{G}{\Rightarrow} w_1 B_1 w_2 \dots w_{n-1} B_{n-1} w_n\,,$$

with $w_1, \dots, w_n \in T^*$, $B_1, \dots, B_{n-1} \in N$.

We know that such clauses are expressible by a Σ_0 formula, so we have a Σ_0 formula $\varphi(x)$ which defines words derivable in $\underline{G}$. Moreover the following is provable in $I\Sigma_0$:

(5) $\varphi(x) \to x$ is of the form (g.2) for some rule (g.3);
(6) $\varphi(\underline{B}_1[v_1]) \,\&\, \dots \varphi(\underline{B}_{n-1}[v_{n-1}]) \to \varphi(\underline{A}[w_1\underline{B}_1[v_1]w_2 \dots$
$\dots w_{n-1}\underline{B}_{n-1}[v_{n-1}]w_n])$,
for each rule (g.3) of G.

Condition (5) follows immediately from (1), (2) and (4); (6) can be proved in the same way as Lemma 3.34.

Next we need the function which maps words of $(N \cup \underline{T})^+$ on words of $(N \cup T)^*$ so that it only omits the letters which do not belong to $N \cup T$ (* denotes the set of all words, including the empty word). A Σ_0 definition of this function can be constructed using the same idea as we used for substitution. (In fact this function is the substitution of empty words for each $\underline{A}$, $A \in N$ and for each bracket [and].) Let us denote this function by *Forget*(x). It is provable in $I\Sigma_0$ that *Forget* is a homomorphism of the free semigroup $(N \cup \underline{T})^+$ into $(N \cup T)^*$. In particular, it is provable in $I\Sigma_0$ that:

(7) $Forget(\underline{A}[w_1\underline{B}_1[v_1]w_2 \dots w_{n-1}\underline{B}_{n-1}[v_{n-1}]w_n])$
$= w_1 Forget(\underline{B}_1[v_1])w_2 \dots w_{n-1} Forget(\underline{B}_{n-1}[v_{n-1}])w_n.$

Recall that we do not allow rules of the form (A, B), for $A, B \in N$. Thus in (g.3) either $n \geq 3$ or at least one w_i is nonempty. This together with (7) can be used to show in $I\Sigma_0$ that:

(8) $\varphi(x) \to lh(x) \leq \overline{7} * lh(Forget(x))$,

which implies a polynomial bound to x in terms of $Forget(x)$.

Now we can define our formulae φ_A:

(9) $\varphi_A(x) \equiv (\exists y)(x = Forget(\underline{A}y) \,\&\, \varphi(\underline{A}y))$.

Hence by (8) these formulae are Σ_0 over $I\Sigma_0$. The rest of the proof is easy. To prove (g.1), first suppose $\varphi_A(x)$ in $I\Sigma_0$. Then, by (9), for some y and A, $x = Forget(\underline{A}y)$ and $\varphi(\underline{A}y)$. By (5), $\underline{A}y$ is of the form (g.2). Hence by (7)

$$x = w_1 y_1 w_2 \ldots w_{n-1} y_{n-1} w_n ,$$

Where

$$y_i = Forget(\underline{B_i}[v_i]) , \quad i = 1, \ldots, n-1 .$$

Thus, by definition (9), $\varphi_{B_i}(y_i)$ for $i = 1, \ldots, n-1$. To prove the converse implication, suppose that

$$x = w_1 y_1 w_2 \ldots w_{n-1} y_{n-1} w_n ,$$

and

$$\varphi(\underline{B_1}[y_1]) \,\&\, \ldots \varphi(\underline{B_{n-1}}[y_{n-1}]) ,$$

for some rule (g.3). Applying (9) and (7) we get $x = Forget(\underline{A}y)$, for $\underline{A}y$ of the form (g.2). Then, by (6), we have also $\varphi(\underline{A}y)$, thus $\varphi_A(x)$ holds true. □

3.38 Corollary. $A \overset{*}{\Rightarrow} w$ iff $\varphi_A(w)$ is true. Hence context-free languages are in Σ_0^N.

Proof. By induction on the length of w. (Repeated application of (g.1) produces a syntactical analysis of w.) □

This corollary is only a by-product. The meaning of the theorem is that it not only produces a Σ_0 formula, but also it gives inductive clauses which determine the concept in $I\Sigma_0$ in a similar way as (c.1–2) determine exponentiation and (e.1–3) determine function *nuon*.

As our first example, we consider the definition of numerals. We cannot use

$$S^n(\overline{0}) = \underbrace{SSS \ldots S}_{n \text{ times}}(\overline{0}) ,$$

in $I\Sigma_0$, since the code of such an expression is exponential in n, thus $I\Sigma_0$ does not prove that $S^n(\overline{0})$ exists for every n. In $I\Sigma_0$ we use *dyadic numerals*

which are defined by

$$\overline{0} = \overline{0}, \quad \overline{1} = S(\overline{0}),$$
$$\overline{2n} = (\overline{1}+\overline{1})*(\overline{n}), \quad \overline{2n+1} = (\overline{1}+\overline{1})*(\overline{n})+\overline{1}, \quad \text{for } n>0.$$

Using a context-free grammar G we first formalize positive numerals. G has one nonterminal S_0, terminals $\overline{1}, +, *, (,)$ and rules

$$S_0 \Rightarrow \overline{1} \mid (\overline{1}+\overline{1})*(S_0) \mid (\overline{1}+\overline{1})*(S_0)+\overline{1},$$

(this is the standard way of writing three rules at once). Let $\varphi_{S_0}(x)$ be the formula given by Theorem 3.37. Then $I\Sigma_0$ proves

$$\varphi_{S_0}(x) \equiv x = \overline{1} \vee (\exists y)(\varphi_{S_0}(y) \,\&\, x = \text{`}(\overline{1}+\overline{1})*(y)\text{'}) \vee (\exists y)(\varphi_{S_0}(y) \,\&\, x = \text{`}(\overline{1}+\overline{1})*(y)+\overline{1}\text{'}),$$

where we use '...' to denote a string of characters. Hence, if we define

$$Num(x) \equiv x = \overline{0} \vee \varphi_{S_0}(x),$$

then $Num(x)$ satisfies natural clauses which determine this concept. The value $val(x)$ of a numeral x is easily Σ_0 definable, since the binary expansion of $val(\overline{n})$ is encoded in $\overline{n}$. Thus the function $num(x)$, the x-th numeral, is also Σ_0 definable, namely by

$$num(x) = y \equiv Num(y) \,\&\, val(y) = x.$$

In logic we usually reduce the definition of such a concept to terms and thus we do not have to use the full generality of Theorem 3.37. Therefore our second example will be the so called *Polish notation*, which is notation without brackets, thus we cannot reduce it directly to the concept of terms. Let us define variable-free formulae in the language consisting of $\overline{0}$, $\overline{1}$ – constants, **add**, **mult** – binary operations, $=$ – a binary predicate, **non** – a unary connective, **et**, **vel** – binary connectives. The grammar will have two nonterminals S_0, S_1, the rules are

$$S_0 \Rightarrow {=}S_1S_1 \mid \mathbf{non}S_0 \mid \mathbf{et}S_0S_0 \mid \mathbf{vel}S_0S_0,$$
$$S_1 \Rightarrow \overline{0} \mid \overline{1} \mid \mathbf{add}S_1S_1 \mid \mathbf{mult}S_1S_1.$$

The meaning of S_0 is "formula", the meaning of S_1 is "term". Theorem 3.37 gives us a Σ_0 formula defining variable-free formulae such that the inductive clauses (given by the rewriting rules) are provable in $I\Sigma_0$.

*

We have developed several tools for defining syntactical concepts in $I\Sigma_0$. Their application to the definition of terms and formulae in the full language of arithmetic, to the definition of proofs etc. is quite routine now, so we can end this section here.

4. Witnessing Functions

(a) Introduction

Suppose a sound theory T proves a sentence $(\forall x)(\exists y)\varphi(x,y)$. If T is weak and φ is simple, then we have some additional information on the functions which *witness* the existential quantifier in this sentence, i.e. functions f for which we have $(\forall x)\varphi(x,f(x))$. A classical result says that if T is $I\Sigma_1$ and φ is bounded, then there is such an f which is *primitive recursive*, see Theorem IV.3.6. Parikh [Parikh 71] has noted that f is linear time computable if T is $I\Sigma_0$ and φ is bounded. Buss [Buss 86, Bounded Arith.] has defined a theory S_2 (a conservative extension of $I\Sigma_0 + \Omega_1$), fragments S_2^i and subsets Σ_i^b of bounded formulae in the extended language. His theorems characterize the classes of functions in the polynomial hierarchy using the above phenomenon. In particular, if φ is a Σ_1^b formula and T is S_2^1, then f is polynomial time computable. The converse of this special case is also true: if f is polynomial-time computable, then there exists a Σ_1^b formula which defines f and for which it is provable in S_2^1 that f is total.

We start this section with the definitions of these classes of formulae and fragments of bounded arithmetic. The next step will be positive results about definability of functions of certain complexity in these fragments. It turns out that this is quite important information about the fragments. For instance, knowing that T_2^i defines functions of $\square_{i+1}^p$, for $i \geq 1$, it is easy to prove Buss' theorem for T_2^i. The main part is subsection (d) where witnessing theorems of Buss and of Krajíček and Takeuti are proved. In subsection (e) we shall show a reduction of the problem of finite axiomatizability of bounded arithmetic to the problem of collapsing the polynomial time hierarchy.

Since the fragments of bounded arithmetic have been extensively studied in [Buss 86, Bounded Arith.], we concentrate on results and proofs not included in that book. In particular, as Buss uses proof theory, we shall give proofs of the witnessing theorems based on model theory. We shall refer to Buss's book for basic facts provable in these fragments; however the readers with some training in bounded arithmetic, for example those who have worked through Sect. 3, should be able to make their own proofs without consulting that book.

(b) Fragments of Bounded Arithmetic

We shall extend the usual language of arithmetic L_0 by adding two unary function symbols $|x|$ and $\llcorner x/2 \lrcorner$, and one binary operation $x \# y$. The intended interpretation of $|x|$ is

$$|x| = \lceil \log_2(x+1) \rceil = \text{the length of the binary expansion of } x \text{ if } x > 0, \text{ otherwise } 0,$$

i.e. the same as in Sect. 3. Here $\lceil y \rceil$ denotes the least integer $z \geq y$; $\llcorner y \lrcorner$ denotes the integer part of y, thus the interpretation of $\llcorner x/2 \lrcorner$ is clear. The intended interpretation of the *smash function* is

$$x \# y = 2^{|x|*|y|} = \text{the power of 2 whose binary expansion has length } |x| * |y| + 1.$$

Observe that the set of the lengths of numbers in a sound arithmetical theory containing the smash function is closed under multiplication, hence under any polynomial increase (for standard polynomials). This is important, since it enables us to formalize many standard constructions, in particular polynomial time computations. We shall denote this extension of L_0 by L_2. In this section we shall work in the language L_2 and in an extension $\hat{L}_2$ of L_2.

Now we recall the basic system of open axioms for the extended language. It is called simply *BASIC* and it plays a similar role to Q in the usual language.

4.1 Definition. *BASIC* is the following theory:

(1) $y \leq x \to y \leq Sx$;
(2) $x \neq Sx$;
(3) $\overline{0} \leq x$;
(4) $x \leq y \,\&\, x \neq y \equiv Sx \leq y$;
(5) $x \neq \overline{0} \to \overline{2}x \neq \overline{0}$;
(6) $y \leq x \vee x \leq y$;
(7) $x \leq y \,\&\, y \leq x \to x = y$;
(8) $x \leq y \,\&\, y \leq z \to x \leq z$;
(9) $|\overline{0}| = \overline{0}$;
(10) $x \neq \overline{0} \to |\overline{2}x| = S(|x|) \,\&\, |S(\overline{2}x)| = S(|x|)$;
(11) $|\overline{1}| = \overline{1}$;
(12) $x \leq y \to |x| \leq |y|$;
(13) $|x \# y| = S(|x| * |y|)$;
(14) $\overline{0} \# y = \overline{1}$;
(15) $x \neq \overline{0} \to 1 \# (\overline{2}x) = \overline{2}(1 \# x) \,\&\, 1 \# (S(\overline{2}x)) = \overline{2}(1 \# x)$;
(16) $x \# y = y \# x$;
(17) $|x| = |y| \to x \# z = y \# z$;
(18) $|x| = |u| + |v| \to x \# y = (u \# y) * (v \# y)$;
(19) $x \leq x + y$;
(20) $x \leq y \,\&\, x \neq y \to S(\overline{2}x) \leq \overline{2}y \,\&\, S(\overline{2}x) \neq \overline{2}y$;
(21) $x + y = y + x$;
(22) $x + \overline{0} = x$;
(23) $x + Sy = S(x + y)$;

(24) $(x+y)+z = x+(y+z)$;
(25) $x+y \leq x+z \equiv y \leq z$;
(26) $x * \overline{0} = \overline{0}$;
(27) $x * (Sy) = xy + x$;
(28) $xy = yx$;
(29) $x(y+z) = xy + xz$;
(30) $x \geq \overline{1} \to (xy \leq xz \equiv y \leq z)$;
(31) $x \neq \overline{0} \to |x| = S(|\llcorner x/2 \lrcorner|)$;
(32) $y = \llcorner x/2 \lrcorner \equiv (\overline{2}y = x \vee S(\overline{2}y) = x)$,
where $\overline{1}$ and $\overline{2}$ are numerals as defined in Sect. 3.

This system is probably not quite optimal. One can play with it in order to find a shorter or nicer system. For instance, if we redefine

$$|x| = \begin{cases} \lceil \log_2(x+1) \rceil - 1 & \text{if } x > 0, \\ 0 & \text{otherwise,} \end{cases}$$

then $x \# y = 2^{|x|*|y|}$ becomes associative and distributive with respect to multiplication, hence one would obtain a more homogeneous axiomatic system for such functions. We shall not do it here, as we want to concentrate on more interesting questions.

We shall use the bounded quantifiers $(\forall x \leq \tau)$ and $(\exists x \leq \tau)$; note that τ is a term in the language L_2, not just in L_0 (not containing x, of course). Moreover we also need the so-called *sharply bounded quantifiers*. They have the form

$$(\forall x \leq |\tau|) \text{ and } (\exists x \leq |\tau|),$$

i.e. the outermost function in the bounding term is the length function. Their relation to ordinary bounded quantifiers in bounded arithmetic is similar to the relation of bounded quantifiers to unbounded quantifiers in *PA*. The classes Σ_i^b and Π_i^b of bounded formulae are defined as follows.

4.2 Definition.

(1) $\Pi_0^b = \Sigma_0^b$ consist of formulae with sharply bounded quantifiers only;
(2) Σ_{i+1}^b and Π_{i+1}^b are the least sets satisfying
  (a) $\Sigma_i^b, \Pi_i^b \subseteq \Sigma_{i+1}^b$, and $\Sigma_i^b, \Pi_i^b \subseteq \Pi_{i+1}^b$;
  (b) $\alpha \in \Sigma_{i+1}^b \Rightarrow (\exists x \leq \tau)\alpha \in \Sigma_{i+1}^b, (\forall x \leq |\tau|)\alpha \in \Sigma_{i+1}^b$,
  $\alpha \in \Pi_{i+1}^b \Rightarrow (\forall x \leq \tau)\alpha \in \Pi_{i+1}^b, (\exists x \leq |\tau|)\alpha \in \Pi_{i+1}^b$;
  (c) $\alpha, \beta \in \Sigma_{i+1}^b \Rightarrow \alpha \,\&\, \beta, \alpha \vee \beta \in \Sigma_{i+1}^b$,
  $\alpha, \beta \in \Pi_{i+1}^b \Rightarrow \alpha \,\&\, \beta, \alpha \vee \beta \in \Pi_{i+1}^b$;
  (d) $\alpha \in \Sigma_{i+1}^b, \beta \in \Pi_{i+1}^b \Rightarrow \neg\beta, \beta \to \alpha \in \Sigma_{i+1}^b$,
  $\alpha \in \Pi_{i+1}^b, \beta \in \Sigma_{i+1}^b \Rightarrow \neg\beta, \beta \to \alpha \in \Pi_{i+1}^b$.

(3) α is in Δ_i^b with respect to a theory T, if α is equivalent to a Σ_i^b formula and a Π_i^b formula in T.
(4) $\Sigma_0^b(\Sigma_i^b)$ is defined as the closure of Σ_i^b under connectives and sharply bounded quantification; note that this class is sometimes denoted also by $\Sigma_{i+1}^b \cap \Pi_{i+1}^b$.

Later on we shall see that, for $i \geq 1$, Σ_i^b (resp. Π_i^b) formulae define just the sets in Σ_i^p (resp. Π_i^p). The most important formalizations of induction for bounded formulae are the usual schema of induction and the following ones.

4.3 Definition.

(1) $PIND(\alpha(x))$ is the formula

$$\alpha(\overline{0}) \,\&\, (\forall x)(\alpha(\llcorner x/2 \lrcorner) \to \alpha(x)). \to (\forall x)\alpha(x);$$

(2) $LIND(\alpha(x))$ is the formula

$$\alpha(\overline{0}) \,\&\, (\forall y < |x|)(\alpha(y) \to \alpha(S(y))). \to \alpha(|x|);$$

(3) $LMIN(\alpha(x))$ is the formula

$$(\exists x)\alpha(x) \to \alpha(\overline{0}) \vee (\exists x)(\alpha(x) \,\&\, (\forall y \leq \llcorner x/2 \lrcorner)(\neg\alpha(x)));$$

(4) $PIND\Gamma$ ($LIND\Gamma$, $LMIN\Gamma$, respectively) is the schema (or the set) $PIND(\alpha(x))$ ($LIND(\alpha(x))$, $LMIN(\alpha(x))$, respectively) for $\alpha(x) \in \Gamma$.

Here we assume that $\alpha(x)$ may contain other free variables, which are called parameters. We have written $LIND$ in the form which contains only one additional quantifier and this quantifier is sharply bounded. Thus if α is Δ_i^b, then $LIND(\alpha(x))$ is also Δ_i^b; if $\alpha(x)$ is in $\Pi_i^b \cup \Sigma_i^b$, then $LIND(\alpha(x))$ is $\Sigma_0^b(\Sigma_i^b)$.

Now we are ready to define the hierarchy of subtheories of bounded arithmetic.

4.4 Definition.

(1) For $i \geq 0$, S_2^i is $BASIC + PIND\Sigma_i^b$.
(2) For $i \geq 0$, T_2^i is $BASIC + I\Sigma_i^b$.
(3) S_2 is $\bigcup_i S_2^i$; T_2 is $\bigcup_i T_2^i$.

The most important fragment of S_2 is S_2^1. It serves as a basic theory modulo which results on fragments of S_2 are proved. Its role is similar to

the role of $I\Sigma_1$ in *PA*. In particular we shall show in the next subsection that all polynomial time computable functions are definable and provably total in S^1_2. As for weaker fragments: S^0_2 is too weak (it does not prove the existence of the predecessor, see [Takeuti, Sharply Bounded]); the strength of T^0_2 is not quite known. We shall not repeat the formalization of elementary mathematical concepts in S^1_2 which has been done in [Buss 86, Bounded Arith.]. One can use some techniques from Sect. 3, mainly those which were mentioned in connection with $I\Sigma_0 + \Omega_1$. (Those which reduced the size of the quantified numbers by using more quantifiers are useless for S^1_2, since we need Σ^b_1 definitions.) Here we briefly mention a definition of coding in S^1_2.

First we re-define $Pow(x)$ (*x is a power of* 2) by an open formula in L_2:

$$Pow(x) \equiv \overline{2}x = x \# \overline{1}.$$

In order to see that this works, we shall show in *BASIC* that Pow satisfies the following recursive condition:

$$Pow(x) \to (x = \overline{1} \vee (\exists y \leq x)(x = \overline{2}y \,\&\, Pow(y))).$$

One can easily show $\neg Pow(\overline{0})$. Suppose $x > 1$. Then, by axiom (32) of *BASIC*, $x = \overline{2}y$ or $x = S(\overline{2}y)$, for $y = \llcorner x/2 \lrcorner$ and $y > 0$. By the definition of Pow and (15), we have $\overline{2}x = x \# \overline{1} = \overline{2}(y \# \overline{1})$, hence (by (1), (6), (7) and (30)) we have $x = y \# \overline{1}$. Since $y > 0$, we get from (15) that x is even. Hence we must have the first possibility $x = \overline{2}y$ (*BASIC* proves that $S(\overline{2}u) \neq \overline{2}v$). Thus $\overline{2}y = x = y \# \overline{1}$, i.e. $Pow(y)$.

Now we can re-define $bit(x,i)$ (*the i-th digit of x*) by a Σ^b_1 formula:

$$\begin{aligned} bit(x,i) = z \equiv z \leq 1 \,\&\, (\exists y, u_2 \leq x)(\exists u_1 < y)(Pow(y) \,\&\, |y| = i \\ \&\, x = \overline{2}u_2 y + zy + u_1), \text{ if } i > 0, \\ = \overline{0} \text{ otherwise}. \end{aligned}$$

Since the decomposition above is unique (see Theorem I.1.30), we have also a Π^b_1 definition:

$$\begin{aligned} bit(x,i) = z \equiv (\forall w \leq 1)(\forall y, u_2 \leq x)(\forall u_1 < y)(Pow(y) \,\&\, |y| = i \\ \&\, x = \overline{2}u_2 y + wy + u_1 \to w = z), \text{ if } i > 0, \\ = \overline{0} \text{ otherwise}. \end{aligned}$$

Thus we have a Δ^b_1 definition of *bit* in S^1_2. Similarly we can define function *sgm*, (see 3.18). The function *bit* codes 0-1 sequences. If we want to code arbitrary sequences, we can proceed as follows. A sequence will be a triple $p = (u,v,w)$ (more precisely $((u,v),w)$, thus every number is a sequence),

where u is the code of the sequence, v is its *maximal element* and w is the *length of the sequence*, i.e.

$$\max(p) = v\,,$$
$$lh(p) = w\,.$$

The function *the i-th element of p* is defined by

$$(p)_i = x \equiv x \leq v \,\&\, (\forall j \leq |v|)(bit(x,j) = bit(u, iv + j)),\ \text{if } i < w\,,$$
$$(p)_i = \overline{0} \text{ otherwise}.$$

Hence all these functions are Δ_1^b. Finally we observe that we can bound u by

$$|u| \leq |v|.w\,.$$

Thus for some term *bound* we have

$$p \leq bound(\max(p), 2^{lh(p)})\,. \tag{a.1}$$

4.5 Proposition.
(1) For $i \geq 1$, $S_2^i \equiv S_2^1 + PIND\Pi_i^b \equiv S_2^1 + LIND\Sigma_i^b \equiv S_2^1 + LIND\Pi_i^b$.
(2) For $i \geq 1$, $T_2^i \equiv S_2^1 + I\Pi_i^b$.

Proof. We shall show only $S_2^i \equiv S_2^1 + LIND\Sigma_i^b$; the proofs of the remaining equivalences are the same as for fragments of *PA*, see Theorem I.2.4.

First we shall derive $LIND\Sigma_i^b$ in S_2^i. Let $\varphi(x)$ be Σ_i^b. Let $\psi(x)$ be $\varphi(|x|)$. Then by (10) and (32) of *BASIC*, the inductive clause of $LIND(\varphi(x))$

$$\varphi(x) \to \varphi(S(x))$$

implies the inductive clause of $PIND(\psi(x))$

$$\psi(\llcorner y/2 \lrcorner) \to \psi(y)\,.$$

Thus $LIND\Sigma_i^b$ is reduced to $PIND\Sigma_i^b$.

Now we shall derive $PIND\Sigma_i^b$ in $S_2^1 + LIND\Sigma_i^b$. Let $\varphi(x)$ be a Σ_i^b formula for which we want to prove *PIND*. Thus we assume $\varphi(\overline{0})$ and $(\forall x)(\varphi(\llcorner x/2 \lrcorner) \to \varphi(x))$. Let $msp(x,y)$ be the number whose binary representation consists of the first y digits of x (and, say, it is x if $y > |x|$). This function has a Δ_1^b definition in S_2^1 (use the function *sgm*). The idea of the proof is to show, using *LIND*, that all end segments of x satisfy φ, hence also x itself satisfies φ. Formally, we apply *LIND* to $\varphi(msp(x,y))$ with y as the induction variable and x as a parameter. Thus $PIND(\varphi(x))$ is reduced to

$LIND(\varphi(msp(x,y)))$ since $msp(x,0)=0$, $\llcorner msp(x,S(y))/2\lrcorner = msp(x,y)$ and $msp(x,|x|)=x$. □

4.6 Theorem. For $i \geq 1$, $S_2^i \vdash I\Delta_i^b$.

Proof. Recall the meaning of $I\Delta_i^b$: for every $\alpha \in \Sigma_i^b$ and $\beta \in \Pi_i^b$

$$(\forall x)(\alpha(x) \equiv \beta(x)) \to I(\alpha(x))\,.$$

The idea of the proof is the following. The schema of induction tells us that we cannot define a proper subset of numbers which contains 0 and is closed under the successor. The schema *PIND* implies that there is no subset of numbers containing 0 and closed under the successor *and the function* $2x$. We already know (Theorem II.3.5) that for every cut we can define a subcut closed under addition, hence also closed under $2x$. We shall use this construction to reduce induction to *PIND*. There is a small technical complication, since we have to use only bounded formulae, while the original construction uses unbounded quantifiers.

Let α and β be given. Put

$$\psi(y,a) \equiv_{df} (\forall x \leq a)(\alpha(x) \to \beta(\min(x+y,a)))\,.$$

Thus ψ is Π_i^b. Working in S_2^i, assume $\alpha(x) \equiv \beta(x)$, $\alpha(0)$ and $(\forall x)(\alpha(x) \to \alpha(S(x)))$. Furthermore assume that $\alpha(a)$ fails for some a. We have $\psi(0,a)$ trivially and $\psi(y,a) \to \psi(S(y),a)$, since $(\forall x)(\alpha(x) \to \alpha(S(x)))$. We shall prove $\psi(y,a) \to \psi(2y,a)$. Assume $\psi(y,a)$ and $\alpha(x)$ for some $x \leq a$. By $\psi(y,a)$ we have $\alpha(\min(x+y,a))$. Thus $x+y < a$, since $\neg\alpha(a)$. Hence $\alpha(x+y,a)$. Applying $\psi(y,a)$ once again (with $x+y$ instead of x) we get $\alpha(\min(x+2y,a))$. Hence we have $\psi(y,a) \to \psi(2y,a)$. Since

$$x = 2\llcorner x/2\lrcorner \vee x = S(2\llcorner x/2\lrcorner)\,,$$

the formula

$$(\forall y)(\psi(y,a) \to \psi(S(y),a)\,\&\,\psi(\overline{2}y,a))$$

implies

$$(\forall y)(\psi(\llcorner y/2\lrcorner,a) \to \psi(y,a))\,.$$

Hence using $PIND(\psi)$ we get $\psi(a,a)$. Since $\alpha(0)$, this implies $\alpha(a)$. We have derived $\alpha(a)$ from its negation, hence it must be true. Thus we have shown induction for α. □

Note that, for $i \geq 2$, a weaker version of this theorem can be derived also from Corollary 4.28 below using the Σ_{i+1}^b conservativity of S_2^{i+1} over T_2^i (Corollary 4.34).

4.7 Theorem.
(1) For $i \geq 0$, $S_2^i \subseteq T_2^i \subseteq S_2^{i+1}$;
(2) $S_2 \equiv T_2$ and they are conservative extensions of $I\Sigma_0 + \Omega_1$.

Proof. (1) To prove the first inclusion, observe that T_2^i trivially proves $LIND\Sigma_i^b$ and apply Theorem 4.5(1). The second inclusion is a corollary of Theorem 4.6.

(2) The equivalence of theories S_2 and T_2 follows from (1) of the preceding proposition. In order to show that $I\Sigma_0+\Omega_1$ is contained in S_2 we have only to prove axiom Ω_1 in S_2, which follows easily from the bound $\omega_2(x) \leq (x \# x)^4$.

Let us prove the conservativity of S_2 over $I\Sigma_0+\Omega_1$. We already know that $\llcorner x/2 \lrcorner$ and $|x|$ are definable in $I\Sigma_0 + \Omega_1$. As we have a Σ_0 definition of the exponentiation relation, we can define the graph of the function $x \# y$. This function is provably total in $I\Sigma_0+\Omega_1$, because of the bound $x \# y \leq \omega_2(x+y)$. The axioms of $BASIC$ are easy too. To prove induction for bounded formulae in language L_2 in $I\Sigma_0 + \Omega_1$, recall that by Proposition 1.3 we can eliminate terms as bounds in the formulae used in induction. Then we use the definitions of the new operations to translate the formulae into the language L_0 and apply $I\Sigma_0$. □

4.8 Theorem. (a) For $i \geq 1$, and $\alpha(u,x) \in \Sigma_0^b(\Sigma_i^b)$,

$$S_2^i \vdash (\exists w)(\forall x < |y|)(\alpha(u,x) \equiv bit(w, x + \overline{1}) = \overline{1}).$$

(b) For $i \geq 1$, $S_2^i \vdash LIND(\Sigma_0^b(\Sigma_i^b))$.

Note that (a) is a version of the $\Sigma_0^b(\Sigma_i^b)$ comprehension schema in S_2^i where we use $bit(w,x)$ as a convenient coding of 0-1 sequences.

Proof. (a) Consider a more general schema for $\alpha \in \Sigma_0^b(\Sigma_i^b)$:

$$\text{(b.1)}\quad S_2^i \vdash (\exists w)(\forall x_1 < |y_1|)\ldots(\forall x_n < |y_n|)(\alpha(u, x_1, \ldots, x_n) \equiv bit(w, \overline{1} + (x_1, (x_2, \ldots, x_n))) = \overline{1}).$$

We should show that (b.1) holds for $\alpha \in \Sigma_i^b$ and that the set of formulae α satisfying (b.1) is closed under boolean operations and sharply bounded quantifications.

Suppose $\alpha \in \Sigma_i^b$. Take the formula $\beta(z,u)$ defined by

$$(\exists w)(|w| \leq |y_1| * |y_2| * \ldots * |y_n| \,\&\, nuon(w) = z \,\&\, (\forall x_1 < |y_1|)\ldots(\forall x_n < |y_n|)(bit(w, \overline{1} + (x_1, (x_2, \ldots, x_n))) = \overline{1} \rightarrow \rightarrow \alpha(u, x_1, \ldots, x_n))).$$

Let us work in S_2^i. This formula is true for $z = 0$ and fails for $z = |y_1| * |y_2| * \cdots * |y_n| + 1$, since $nuon(w) \leq |w|$. Since α is Σ_i^b, $nuon$ is Δ_1^b and we can bound w, say by $y_1 \# y_2 \# \ldots \# y_n$, the formula β is also Σ_i^b. Hence we can use $LIND\Pi_i^b$ to find the smallest z for which $\beta(z,u)$ fails. Then take w such that $\beta(z-1,w)$. If

$$z - 1 = |y_1| * |y_2| * \cdots * |y_n|,$$

then all bits of w are 1, hence we have (b.1) for this w. Now suppose that

$$z - 1 < |y_1| * |y_2| * \cdots * |y_n|.$$

If (b.1) were not true for this w, then we could add another 1 in the binary representation of w which codes an instance of $\alpha(u, x_1, x_2, \ldots, x_n)$. But this would contradict the condition that $z-1$ is the last parameter for which $\beta(z-1,w)$ is true. Hence w codes exactly the set determined by α.

The case of propositional connectives and sharply bounded quantifiers is easy.

(b) follows immediately from (a). □

In Bounded Arithmetic there is a counterpart of the *collection axiom* for Peano Arithmetic, see I.2.1. We shall call it *the bounded collection axiom.* Sometimes it is also called *replacement axiom.*

4.9 Definition. (i) $BB(\alpha(x,y))$ *bounded collection* is the following formula

$$(\forall x \leq |t|)(\exists y \leq s)\alpha(x,y). \rightarrow (\exists w)(\forall x \leq |t|)((w)_x \leq s \,\&\, \alpha(x,(w)_x)).$$

(ii) $BB\Gamma$ is the schema $BB(\alpha)$ for $\alpha \in \Gamma$.

Using (a.1) above, we can bound the variable w in the bounded collection axiom; thus it has the following form (equivalent in S_2^1):

$$(\forall x \leq |t|)(\exists y \leq s)\alpha(x,y). \rightarrow \\ \rightarrow (\exists w \leq \tau(s,t))(\forall x \leq |t|)((w)_x \leq s \,\&\, \alpha(x,(w)_x)),$$

where τ is a suitable term ($\tau(s,t) = bound(s, 2^{|t|+1})$).

4.10 Theorem. For $i \geq 1$, $S_2^i \vdash BB\Sigma_i^b$.

Proof. Let $\alpha(x,y)$ be a Σ_i^b formula. Assume

$$(\forall x \leq |t|)(\exists y \leq s)\alpha(x,y).$$

Then we obtain

$$(\exists w \leq \tau(s,t))(\forall x \leq |t|)((w)_x \leq s \,\&\, \alpha(x,(w)_x))$$

by

$$LIND((\exists w \le \tau(s,t))(\forall x \le |z|)((w)_x \le s \,\&\, \alpha(x,(w)_x)))\,,$$

where the induction variable is z. The formula is Σ^b_i, since coding is Δ^b_1. We use the elementary facts that the empty sequence has a code, each sequence can be extended by an arbitrary element and the estimate of the size of a sequence. □

Note that the implication converse to that above is already provable in S^1_2 (using only the fact that $(w)_x$ is a definable function in S^1_2). The bounded form of the bounded collection axiom enables us to interchange sharply bounded quantifiers with bounded quantifiers. However we cannot push inside all sharply bounded quantifiers without extending the language, since the coding function is not defined by a term in L_2. Therefore we need to extend the language.

4.11 Definition. (i) The language L_2 augmented with the binary operation $(x)_z$, will be denoted by $\hat{L}_2$; the corresponding classes Σ^b_i, Π^b_i etc. will be denoted by $\hat{\Sigma}^b_i, \hat{\Pi}^b_i$, etc.

(ii) We say that a formula is *strict*$\hat{\Sigma}^b_i$ if it is of the form

$$(\exists x_1 \le t_1)(\forall x_2 \le t_2)\ldots(\mathbf{Q}x_i \le t_i)\varphi\,,$$

where $\mathbf{Q}$ is $\exists$ if i is odd and $\forall$ if i is even, and φ is sharply bounded.

We shall not introduce new notation for theories obtained by extending the language and adding the definition of the coding relation. It will always be clear from the context which theory we mean.

4.12 Corollary. For $i \ge 1$, every Σ^b_i formula is equivalent to a *strict*$\hat{\Sigma}^b_i$ formula, provably in S^i_2 (augmented with the definition of $(x)_z$).

Proof. We can shift all sharply bounded quantifiers beyond bounded (but not sharply) quantifiers using the bounded collection schema. Successive quantifiers of the same kind can be merged using the coding function. □

4.13 Corollary. For $i \ge 1$ and $\alpha(x,y)$ in Σ^b_i,

$$S^i_2 \vdash (\exists w)(\forall x \le |t|)[(\exists y \le s)\alpha(x,y) \equiv ((w)_x \le s \,\&\, \alpha(x,(w)_x))]\,.$$

(This is sometimes called *strong replacement*.)

Proof. Let $\alpha(x,y)$ in Σ^b_i be given. By Theorem 4.8, we can code the set of x's such that $(\exists y \le s)\alpha(x,y)$ using some v and formula $bit(v,x) = 1$. Now take $\alpha'(x,y)$ to be

$$\alpha(x,y) \vee bit(v,x) = 0\,.$$

Then we have $(\forall x \leq |t|)(\exists y \leq s)\alpha'(x,y)$, hence we can apply bounded collection; the sequence w given by it satisfies the clause of the theorem. □

(c) Definability of Turing Machine Computations in Fragments of Bounded Arithmetic

Recall that for $i \geq 1$, $\square_i^p$ denotes the set of functions computable in polynomial time using oracles for Σ_{i-1}^p sets; in particular, $\square_1^p$ are just the polynomial time computable functions, since Σ_0^p oracles are superfluous. In this subsection we shall show that functions from $\square_1^p$ have suitable definitions in S_2^1, and the same holds for $\square_i^p$ and T_2^{i-1}, if $i > 1$. This means that for functions from $\square_i^p$ we have Δ_{i+1}^b definitions in the corresponding theories and that basic properties of the functions are provable in these theories. We shall only show that the defining formulae have appropriate complexity and that they define total functions. Other properties follow from the construction and their proofs are omitted.

We shall consider deterministic Turing machines, possibly with oracles. Let an oracle Turing machine M be given, let e be the code of it, let A be an oracle. We formalize the computation of M on input a as two sequences w and q, where $(w)_i$ is the instantaneous description of the state of M, the position of the heads and the content of the tapes, and $(q)_i$ is the query asked in step i.[1] The computation is determined by a constant *firstq* (let us set it equal to 0) and five functions *firstw*, $nextw_0$, $nextw_1$, $nextq_0$, $nextq_1$ as follows:

$$(w)_0 = firstw(a), \quad (q)_0 = firstq\,;$$
$$(w)_{i+1} = nextw_0((w)_i,(q)_i), \quad (q)_{i+1} = nextq_0((w)_i,(q)_i), \quad \text{if } (q)_i \notin A\,;$$
$$(w)_{i+1} = nextw_1((w)_i,(q)_i), \quad (q)_{i+1} = nextq_1((w)_i,(q)_i), \quad \text{if } (q)_i \in A\,.$$

We can think of $(w)_i$ as a sequence which encodes the current situation on the tapes and encodes the state of M. Then the functions above are only local transformations of these sequences; hence it is not difficult to write down Δ_1^b definitions for them using our Δ_1^b definition of coding. If the oracle A is defined by a formula $\varphi(x)$, we thus get a formula

$$Comp_{M,\varphi(x)}(w,q,a)$$

expressing that (w,q) *is a computation of* M *on input* a. If M is a machine without an oracle, we shall write simply $Comp_M(w,a)$. We shall assume that

[1] W.l.o.g. we can assume that M asks a query in each computation step; if it does not need information it asks some default fixed query.

$(w)_i = (w)_{i+1}$ iff M stops in the i-th step. Thus computations of length n exist for every n, and we have a simple condition for testing the termination. Furthermore we shall assume that *this* $(w)_i$ *is the output of* M. We shall identify oracles with their defining formulae; thus we shall talk about Σ_i^b *oracles* instead of Σ_i^p *oracles*.

4.14 Lemma.

(i) $Comp_M(w,a)$ is Δ_1^b in S_2^1.

(ii) If $\varphi(x)$ is Σ_i^b, $i \geq 1$, then $Comp_{M,\varphi(x)}(w,q,a)$ is $\Sigma_0^b(\Sigma_i^b)$.

(iii) There exists a term σ such that

$$S_2^1 \vdash Comp_{M,\varphi(x)}(w,q,a) \to w,q \leq \sigma(a, lh(w))\,.$$

Proof. (i) and (ii). We have

$$Comp_M(w,a) \equiv (w)_0 = firstw(a)\,\&\,(\forall i < lh(w)-1)((w)_{i+1} = nextw((w)_i))\,,$$

and

$$\begin{aligned} Comp_{M,\varphi(x)}(w,q,a) \equiv\ & (w)_0 = firstw(a)\,\&\,(\forall i < lh(w)-1) \\ & [(\neg\varphi(q)_i\,\&\,(w)_{i+1} = nextw_0((w)_i,(q)_i)\,\&\,(q)_{i+1} = nextq_0((w)_i,(q)_i)) \\ & \vee\,(\varphi(q)_i\,\&\,(w)_{i+1} = nextw_1((w)_i,(q)_i)\,\&\,(q)_{i+1} = nextq_1((w)_i,(q)_i))]\,. \end{aligned}$$

We already know that all the functions used in these formulae are Δ_1^b definable.

(iii) The length of w and q is polynomial in the number of steps of the computation and the length of input, whence we get the bound. □

Now we prove the relation between the classes of formulae Σ_i^b and the classes of sets Σ_i^p which we have promised. Note, however, that the relation is not quite direct for $i = 0$.

4.15 Theorem.

(i) Every set definable by a Σ_0^b formula is in *LogSpace*.

(ii) Every set in P is definable by a Δ_1^b formula.

(iii) For $i \geq 1$, the sets in Σ_i^p (resp. Π_i^p) are just the sets definable by Σ_i^b (resp. Π_i^b) formulae.

("Definable" means here "definable in the standard model".)

Proof. (i) All the functions and predicates of the language of S_2 are *LogSpace* computable and sharply bounded quantification preserves *LogSpace* computations.

(ii) Let X be in P. Let M be a Turing machine which always stops after $p(|a|)$ steps on input a and outputs 0 iff a belongs to X. Then we can define X by

$$(\exists w)(lh(w) \leq p(|a|) \,\&\, Comp_M(w,a) \,\&\, (w)_{p(|a|)} = 0)\,,$$

or by

$$(\forall w)(lh(w) \leq p(|a|) \,\&\, Comp_M(w,a) \to (w)_{p(|a|)} = 0)\,.$$

By Lemma 4.14, the inner parts of the formulae are Δ_1^b; by the same lemma the bound $lh(w) \leq p(|a|)$ implies that w can be bounded by a term in a, hence the formulae are equivalent to Σ_1^b and Π_1^b formulae respectively, thus they are Δ_1^b.

(iii) By Corollary 4.12 every Σ_i^b formula is equivalent to a strict Σ_i^b formula in the language extended by the coding relation; thus it can be written in the form

$$(\exists y_1 \leq t_1)(\forall y_2 \leq t_2)\ldots(Qy_i \leq t_i)\Phi\,,$$

where Φ is sharply bounded in the extended language and where Q is $\exists$ if i is odd and $\forall$ if i is even. Since the coding relation also belongs to P, formula Φ defines a predicate in P. The bounds are just of the order needed for the definition of Σ_i^p and the relations $y_j \leq t_j$ are in P as well. Thus, by Theorem 2.11, each set definable by a Σ_i^b formula is in Σ_i^p.

To prove the converse let X be in Σ_i^p. Thus X is defined by

$$(\exists y_1, |y_1| \leq p_1(|x|))\ldots(Qy_i, |y_i| \leq p_i(|x|))P(x, y_1, \ldots, y_i)\,,$$

where P is in P. By (ii) we can define P by a Δ_1^b formula. Similarly to above we can replace the bounds $|y_j| \leq p_j(|x|)$ by bounds of the form $y_j \leq t_j$ for suitable terms $t_1, \ldots, t_i$. Thus we obtain a Σ_i^b formula defining X. □

4.16 Theorem. (1) Let M be a Turing machine. Then S_2^1 proves that for every input a and every b, there exists a unique computation of M which has $|b|$ steps. Formally:

(i) $S_2^1 \vdash (\exists w)(Comp_M(w,a) \,\&\, lh(w) = |b|)$;

(ii) $S_2^1 \vdash Comp_M(w,a) \,\&\, Comp_M(w',a)$
$\qquad\qquad \&\, lh(w), lh(w') = |b| \to w = w'\,.$

(2) Let $i \geq 1$, let M be an oracle Turing machine and let $\varphi(x)$ be in Σ_i^b. Then T_2^i proves that for every input a and every b, there exists a unique computation of M with oracle $\varphi(x)$ which has $|b|$ steps. Formally:

(i) $T_2^i \vdash (\exists w, q)(Comp_{M,\varphi(x)}(w,q,a) \,\&\, lh(w) = |b|)$;

(ii) $T_2^i \vdash Comp_{M,\varphi(x)}(w,q,a) \,\&\, Comp_{M,\varphi(x)}(w',q',a)$
$\qquad\qquad \&\, lh(w), lh(w') = |b| \to w = w' \,\&\, q = q'\,.$

Proof. Let M and φ be given, φ in Σ_i^b. Let ψ be defined by

$$\begin{aligned}\psi(w,q,a,b) \equiv\ & lh(w) = |b|+1 \,\&\, (w)_0 = firstw(a) \,\&\, (q)_0 = firstq \,\&\, (\forall j < |b|)\\ & [((w)_{j+1} = nextw_0((w)_j,(q)_j) \,\&\, (q)_{j+1} = nextq_0((w)_j,(q)_j))\\ & \vee (\varphi((q)_j) \,\&\, (w)_{j+1} = nextw_1((w)_j,(q)_j)\\ & \quad \&\, (q)_{j+1} = nextq_1((w)_j,(q)_j))]\,.\end{aligned}$$

The meaning of the formula is that w,q is like a computation in which we follow the advice of the oracle if its answer is negative, but not always do we follow it if its answer is positive. This formula is Δ_1^b if $i=0$, and Σ_i^b if $i>0$. Hence we can prove, using respectively $LIND\,\Sigma_1^b$ or $LIND\,\Sigma_i^b$ on b, that for given a,b there exists at least one pair w,q such that $\psi(w,q,a,b)$. Using $I\Sigma_i^b$ we now minimize the number of steps where w and q do not follow the positive advice. Let Ψ be the Σ_i^b formula defined by

$$\begin{aligned}\Psi(u,a,b) \equiv\ & (\exists w,q \le \sigma(a,|b|))(\psi(w,q,a,b) \,\&\, (\forall j < |b|)\\ & [(w)_{j+1} = nextw_0((w)_j,(q)_j) \,\&\, (q)_{j+1} = nextq_0((w)_j,(q)_j) \rightarrow\\ & \qquad \rightarrow bit(u,|b|-j) = 1)]\,,\end{aligned}$$

where σ is the term from Lemma 4.14(iii). For a and b there exists at least one u satisfying $\Psi(u,a,b)$, since for w,q satisfying $\psi(w,q,a,b)$ the existence of u follows by $LIND\,\Delta_1^b$. By $I\Sigma_i^b$ we have the least u for which $\Psi(u,a,b)$ holds (this is the place where we use the full strength of T). Let w,q be the witnesses of $\Psi(u,a,b)$ for this u. We shall show that for this w,q also the implication complementary to the one in ψ holds for all $j<|b|$:

$$\varphi((q)_j) \rightarrow (w)_{j+1} = nextw_1((w)_j,(q)_j) \,\&\, (q)_{j+1} = nextq_1((w)_j,(q)_j)\,.$$

Suppose not, then we have $\varphi((q)_j)$ and

$$(w)_{j+1} = nextw_0((w)_j,(q)_j) \,\&\, (q)_{j+1} = nextq_0((w)_j,(q)_j)\,,$$

for some $j<|b|$. Hence $bit(u,|b|-j)=1$. Take w' and q' of length $|b|$ such that

$$(w')_k = (w)_k \,\&\, (q')_k = (q)_k\,,$$

for $k=0,\ldots,j$,

$$(w')_{j+1} = nextw_1((w)_j,(q)_j) \,\&\, (q')_{j+1} = nextq_1((w)_j,(q)_j)\,,$$

and

$$(w')_{k+1} = nextw_0((w')_k,(q')_k) \,\&\, (q')_{k+1} = nextq_0((w')_k,(q')_k)\,,$$

for $k = j+1, \ldots, |b|-2$. The existence of such w' and q' is again easily provable by $LIND\Sigma_i^b$. Let u' be determined by

$$\begin{aligned} &bit(u', k) = 1, \quad \text{for } k = 1, \ldots, |b| - j - 1, \\ &bit(u', |b| - j) = 0, \\ &bit(u', k) = bit(u, k), \quad \text{for } k = |b| - j + 1, \ldots, |b|. \end{aligned}$$

Then we have $\Psi(u', a, b)$, since u' corresponds to w', q', but $u' < u$, which is a contradiction. Thus we have shown

$$Comp_{T,\varphi(x)}(w, q, a) \,\&\, lh(w) = |b|.$$

To prove uniqueness we apply $LIND\Delta_1^b$ to

$$(\forall j < |b|)((w)_j = (w')_j \,\&\, (q)_j = (q')_j). \qquad \square$$

4.17 Theorem. $\square_1^p$ functions are definable by formulae which are Δ_1^b in S_2^1; for $i > 0$, $\square_{i+1}^p$ functions are definable by formulae which are Δ_{i+1}^b in T_2^i. Furthermore, for each such formula the corresponding theory proves that the formula defines a total function.

Proof. Let $i \geq 1$. Let f be a $\square_{i+1}^p$ function defined by an oracle machine M and a Σ_i^p oracle A. By Theorem 4.15, A can be defined by a Σ_i^b formula $\varphi(x)$. Let $p(x)$ be a polynomial time bound for M. Then (similarly as in Theorem 4.15) $f(a) = b$ can be defined by

$$(\exists w, q \leq \sigma(a, p(|a|)))(Comp_{M,\varphi(x)}(w, q, a) \,\&\, (w)_{p(|a|)} = b),$$

or by

$$(\forall w, q \leq \sigma(a, p(|a|)))(Comp_{M,\varphi(x)}(w, q, a) \to (w)_{p(|a|)} = b),$$

where σ is the term from Lemma 4.14. Using the same lemma to estimate the complexity of $Comp_{M,\varphi(x)}$ and the existence and uniqueness of the computation (Theorem 4.16) we see that it is a Δ_{i+1}^b definition in T_2^i. For $i = 0$ the proof is similar. The fact that the formulae define total functions in the corresponding theories follows from Theorem 4.16. $\square$

In order to be able to use the definitions of functions for other constructions in T, we need more properties of formulae defining the functions. In particular, we shall use the following properties:

(1) if $t(\mathbf{x})$ is a term and ψ is the definition of the corresponding $\square_1^p$ function, then $t(\mathbf{x}) = y$ is equivalent to $\psi(\mathbf{x}, y)$ in S_2^1;

(2) $\square_i^p$ functions are provably closed under composition, i.e. if $\varphi(\mathbf{x}, y)$ defines $f(\mathbf{x})$ and $\psi_i(\mathbf{x}, y)$ define $g_i(\mathbf{x})$, then for some definition $\chi(\mathbf{x}, y)$ of $f(g_1(\mathbf{x}), \ldots, g_n(\mathbf{x}))$ T_2^i (resp. S_2^1, if $i = 1$) proves

$$\chi(\mathbf{x}, y) \equiv (\exists z_1, \ldots, z_n)(\psi_1(\mathbf{x}, z_1) \,\&\, \ldots \,\&\, \psi_n(\mathbf{x}, z_n) \,\&\, \varphi(z_1, \ldots, z_n, y))\,;$$

(3) $\square_i^p$ functions are closed under definitions by cases determined by Σ_{i-1}^b formulae (it is clear what is the corresponding formula);

(4) $\square_i^p$ functions are closed under bounded recursion.

The last condition can be formalized as follows. Let $f(\mathbf{x}), g(z, \mathbf{x}), k(y, \mathbf{x})$ be functions in $\square_i^p$; then the function $h(y, \mathbf{x})$ defined by the following recursion must also belong to $\square_i^p$:

$$h(0, \mathbf{x}) = f(\mathbf{x})\,,$$
$$h(y, \mathbf{x}) = \min(g(h(\llcorner y/2 \lrcorner, \mathbf{x}), \mathbf{x}), k(y, \mathbf{x}))\,.$$

We require that this schema be provable in T_2^i (resp. S_2^1, if $i = 1$) for the defining formulae of f, g and h. To prove these conditions one has only to check that the proofs in the standard model can be carried out in the fragments of Bounded Arithmetic. We omit the proofs since they are not difficult and contain no essentially new ideas.

So far we are able to talk about a single function form some class $\square_i^p$. Later we shall need formalization of Turing machine computations such that we can talk about Turing machines and oracles in the theory, i.e. we want a formula defining the computation which has parameters also for Turing machines and oracles. In the model theoretical language it means that we want to consider also nonstandard Turing machines and oracles.

A simple solution is to take a universal Turing machine, which is explicitly defined and thus has a formalization in S_2^1 by Theorem 4.17. Then the code of an arbitrary Turing machine will be the word which must be written on the input of the universal one in order to simulate it. A natural universal Turing machine simulates each machine M in such a way that the running time is at most polynomially longer than the running time of M. More precisely, the simulation time is bounded by a polynomial in the original running time and the size of M. In particular, if M runs in polynomial time so does the simulation of M.

This was about Turing machines. Oracles can be presented in a similar fashion. Using a universal Turing machine we can represent every predicate $P(x)$ in P in the form $\mu_0(e, x, t_e(x))$ for some fixed Δ_1^b formula μ_0. Here e is the code (the index) of P and t_e is a suitable term determined by the running time of a Turing machine for P. This is still not quite what we need, since the dependence of t_e on e is not explicit. Thus we shall make an additional natural assumption that the predicate with code e is decidable in time n^e.

Then our representation of predicates in P will be of the form

$$\mu_0(e, x, \overline{2}^{|x|^e}).$$

Note that for e standard the existence of $\overline{2}^{|x|^e}$ is provable in S_2^1. For the representation of predicates in Σ_i^b we have only to add quantifiers. Also there is no need to use two indices, one for the machine and one for the oracle. We shall present our conclusions as a theorem.

4.18 Theorem. There are formulae $\mu_0, \nu_0 \in \Delta_1^b$, $\mu_i \in \Sigma_i^b$, $\nu_i \in \Sigma_0^b(\Sigma_i^b)$ for $i \geq 1$, such that

(i) each predicate which is in P, provably in S_2^1, can be represented in S_2^1 as $\mu_0(\overline{e}, x, \overline{2}^{|x|^{\overline{e}}})$, for some numeral $\overline{e}$;

(ii) for $i \geq 1$ and for every $\varphi(x) \in \Sigma_i^b$, there exists an e such that

$$S_2^1 \vdash \varphi(x) \equiv \mu_i(\overline{e}, x, \overline{2}^{|x|^{\overline{e}}});$$

(iii) for every function f which is polynomial time computable, provably in S_2^1, the relation $f(x) = y$ can be represented in S_2^1 by

$$\nu_0(\overline{e}, x, y, \overline{2}^{|x|^{\overline{e}}}), \quad \text{for some numeral } \overline{e};$$

(iv) for $i \geq 1$ and for every function f which is in $\square_{i+1}^p$, provably in T_2^i, the relation $f(x) = y$ can be represented in T_2^i by

$$\nu_i(\mathbf{e}, x, y, \overline{2}^{|x|^{\overline{e}}}), \quad \text{for some numeral } \overline{e}.$$

Let us state (i) more precisely. A predicate P in P is determined by a 0-1 polynomial time computable function f. For this function we have a defining formula $\varphi(x, y)$ which is Δ_1^p and satisfies some basic conditions in S_2^1. Thus $P(x)$ is defined by $\varphi(x, \overline{0})$. The precise statement of (i) is that for some number e

$$S_2^1 \vdash \varphi(x, \overline{0}) \equiv \mu_0(\overline{e}, x, \overline{2}^{|x|^{\overline{e}}}).$$

The statement (iii) should be understood in the same way.

In order to simplify notation we shall define

$$\{e\}_i(x) = y \equiv_{df} \nu_i(e, x, y, \overline{2}^{|x|^e}).$$

We should keep in mind that it is not a bounded formula. However, if we prove the existence of $\overline{2}^{|x|^e}$, then we can work with it as if it were a Δ_{i+1}^b formula.

(d) Witnessing Functions

In this subsection we shall develop the model theory of fragments of S_2 and prove some theorems of the type mentioned at the beginning of this section. We shall use the fact proved above that Turing machine computations are definable in fragments of S_2.

First we shall show that a substructure of a model M of T_2^i that is closed under the functions of $\square_{i+1}^p$ is Σ_i^b elementary in M and is a model of T_2^i. The same holds for substructures of models of S_2^1 and the functions of $\square_1^p$. This is applied to prove witnessing Theorems 4.27 and 4.29 for fragments T_2^i. Then we prove a witnessing Theorem 4.32 for fragments S_2^{i+1}. The theorem has a stronger conclusion than in the original form used by Buss; it states that $(\forall x)\varphi(x, f(x))$ is provable in the weaker theory T_2^i, if $i > 0$. This allows us to prove immediately the well-known conservation result for such pairs of theories. As the model-theoretical proof of Theorem 4.27 is rather difficult, we first prove an auxiliary Theorem 4.31 about extensions of models of S_2^{i+1}.

4.19 Definition. Let M be a structure for the language of S_2, let $A \subseteq M$, $i \geq 0$. Assume that each $\{e\}_i(x)$ defines a function in M. *The* $\square_i^p$ *closure of* A is the set

$$\{\{e\}_i(a) \mid a \in A \,\&\, e \in N\}.$$

We say that A *is* $\square_i^p$ *closed* if A is its own closure.

We want to use the fact that *polynomial time predicates are absolute in every* $\square_1^p$ *closed substructure of* M, a model of S_2^1. As we do not have symbols for all polynomial time predicates in our language, we have to formalize this statement. Before the next definition let us note that every formula can be transformed to an equivalent formula which has only $\&$, $\vee$ and $\neg$ as connectives, and all negations occur only at atomic formulae. What is important is that this transformation preserves the quantifier complexity. We shall say that such a formula is in *negation normal form*.

4.20 Definition. Let $i \geq 0$, let T be S_2^1 if $i = 0$ and T_2^i otherwise. For a formula φ in negation normal form and a model $M \models T$, we define inductively that φ *has* $\square_{i+1}^p$ *Skolem functions in* M:

(1) every open formula has $\square_{i+1}^p$ Skolem functions in M;

(2) if φ and ψ have $\square_{i+1}^p$ Skolem functions in M, then $\forall x\varphi$, $\varphi \,\&\, \psi$ and $\varphi \vee \psi$ have $\square_{i+1}^p$ Skolem functions in M;

(3) if $\varphi(\mathbf{x}, y)$ has $\square_{i+1}^p$ Skolem functions in M, then $(\exists y)\varphi(\mathbf{x}, y)$ has $\square_{i+1}^p$ Skolem functions in M, if for some f in $\square_{i+1}^p$

$$M \models (\forall \mathbf{x})((\exists y)\varphi(\mathbf{x}, y) \to \varphi(\mathbf{x}, f(\mathbf{x}))).$$

Here $f(\mathbf{x})$ is represented by $\{\overline{e}\}_i((x_1, (x_2, \ldots, x_n)\ldots))$, where $e \in N$ and $(x_1, \ldots, x_n) = \mathbf{x}$.

4.21 Lemma. Let T be as above, let $M \models T$, let $K \subseteq M$ be $\Box^p_{i+1}$ closed, and let $\varphi(\mathbf{x})$ have $\Box^p_{i+1}$ Skolem functions in M. Then for $\mathbf{a} \in K$,

$$M \models \varphi(\mathbf{a}) \Rightarrow K \models \varphi(\mathbf{a}).$$

Proof. By induction on the complexity of φ. □

4.22 Corollary. If $\varphi(\mathbf{x})$ and $\neg\varphi(\mathbf{x})$ have $\Box^p_{i+1}$ Skolem functions in $M \models T$, T as above, then $\varphi(\mathbf{x})$ is absolute for $\Box^p_{i+1}$ closed substructures of M, i.e. if A is a $\Box^p_{i+1}$ closed substructure of M and $\mathbf{a}$ is a string of elements of A then

$$M \models \varphi(\mathbf{a}) \quad \text{iff} \quad A \models \varphi(\mathbf{a}).$$

The natural definitions of polynomial time computable predicates have $\Box^p_1$ Skolem functions in M. In particular we shall need that the formula $x = (y)_z$ and its negation have $\Box^p_1$ Skolem functions in M. Now let $P(x)$ be an arbitrary polynomial time computable predicate. Then it can be defined by

$$(\text{d.1}) \qquad (\exists w)(|w| \leq p(|x|) \,\&\, \mathit{Comp}(w, x) \,\&\, (w)_{p(|x|)} = 1).$$

It is clear that w and the existentially quantified numbers in *Comp* can be computed in polynomial time, thus (d.1) has $\Box^p_1$ Skolem functions in M. The same is true for the negation of (d.1), hence (d.1) is absolute. Thus in our formalization, polynomial time predicates are absolute in the situation considered above. The same argument works, of course, for polynomial time computable functions. This enables us to work with polynomial time computable predicates as if they were present in the language.

4.23 Lemma. Let M be a model of S^1_2, let K be a $\Box^p_1$ closed substructure. Then

$$K \models \mathit{LIND\text{-}strict}\hat{\Sigma}^b_i \Rightarrow K \models S^i_2.$$

Proof. First we shall show using *LIND* for $\mathit{strict}\hat{\Sigma}^b_i$ formulae that in K every Σ^b_i formula is equivalent to a $\mathit{strict}\hat{\Sigma}^b_i$ one. Thus we will have $\mathit{LIND}\Sigma^b_i$ in K. This is proved by induction on the number of quantifiers. Several quantifiers of the same kind are replaced by a single one using the coding relation. A sharply bounded universal quantifier is exchanged with the next existential bounded quantifier using an instance of $BB\hat{\Sigma}^b_i$, for which $\mathit{LIND\text{-}strict}\hat{\Sigma}^b_i$ is sufficient (see the proof of Theorem 4.10). We must check that the properties of the coding function that we are using are provable already in $\mathit{LIND\text{-}strict}\hat{\Sigma}^b_i$. We need the following properties:

(1) $(\forall x_0 \dots x_j)(\exists y)((y)_0 = x_0 \,\&\, \dots \,\&\, (y)_j = x_j)$, for every j;

(2) $(\exists s)(lh(s) = 0)$;

(3) $(\forall x, s)(\exists t)(lh(t) = lh(s) + \overline{1} \,\&\, (t)_{lh(s)} = x$
$\&\, (\forall z < lh(s))((t)_z = (z)_z))$;

$((t)_{lh(s)}$ is the last element of the sequence t). Again it is clear that these formulae have $\square_1^p$ Skolem functions in M. Since they are true in M they are true also in K by Lemma 4.22.

Similarly we shall show that $LIND\,\Sigma_i^b$ implies $PIND\,\Sigma_i^b$ in K. We repeat the proof of

$$S_2^1 + LIND\,\Sigma_i^b \vdash PIND\,\Sigma_i^b,$$

(Proposition 4.5 (1)) where we needed from S_2^1 only the following properties of the function msp:

$$msp(a, \overline{0}) = \overline{0};$$
$$S(b) \le |a| \to \llcorner msp(a, S(b))/2 \lrcorner = msp(a, b);$$
$$msp(a, |a|) = a.$$

These formulae are true in M and have $\square_1^p$ Skolem functions, hence they are true in K too. □

4.24 Lemma. Let $i \ge 0$, let T be S_2^1 if $i = 0$ and T_2^i otherwise. Then each $\Sigma_0^b(\Sigma_i^b)$ formula has $\square_{i+1}^p$ Skolem functions in each $M \models T$.

Proof. We shall use induction on the depth of the formula. For open formulae the lemma is true by definition. Consider a Σ_i^b formula of the form

$$(\exists x \le t(\mathbf{a}))\varphi(\mathbf{a}, x),$$

where we assume that $\varphi(\mathbf{a}, x)$ has $\square_{i+1}^p$ Skolem functions. We apply ***binary search*** using Σ_i^b oracle

$$(\exists x)(p \le x \le q \,\&\, \varphi(\mathbf{a}, x))$$

to find an x satisfying $x \le t(\mathbf{a}) \,\&\, \varphi(x, \mathbf{a})$, for a given $\mathbf{a}$. At the beginning we set $p := 0$ and $q := t(\mathbf{a})$. First we ask whether $(\exists x \le t(\mathbf{a}))\varphi(\mathbf{a}, x)$. If it is not true, then the output is, say, 0. Otherwise we continue as follows. In each subsequent step we set either

$$p := p, \quad q := \llcorner 1/2(p+q) \lrcorner, \quad \text{if } (\exists x)(p \le x \le \llcorner 1/2(p+q) \lrcorner \,\&\, \varphi(\mathbf{a}, x))$$

or

$$p := \llcorner 1/2(p+q) \lrcorner, \quad q := q, \quad \text{otherwise}.$$

By Theorem 4.16 the computation of length $|t(\mathbf{a})|$ always exists. In order to prove its correctness we need only to show that in the j-th step of the computation the following two conditions are satisfied:

$$q - p \leq t(\mathbf{a})/2^j ;$$
$$(\exists x)(p \leq x \leq q \,\&\, \varphi(\mathbf{a}, x)) .$$

This is proved by induction on j. Once the computation is given, the condition that we are proving is Σ_i^b, hence we need only $LIND\,\Sigma_i^b$. If the formula has the form

$$(\exists x \leq |t(\mathbf{a})|)\varphi(\mathbf{a}, x) ,$$

where φ is Π_i^b, we proceed similarly, but we use thorough search instead of binary search. The case of the universal bounded quantifier and connectives $\&, \vee$ is trivial. □

4.25 Corollary. Let T be as above, let $M \models T$, and let K be a $\Box_{i+1}^p$ closed substructure of M. Then K is $\Sigma_0(\Sigma_i^b)$ elementary.

Proof. By Lemma 4.21 and Lemma 4.24. □

Since $LIND\,\Sigma_i^b$ is a $\Sigma_0^b(\Sigma_i^b)$ formula, we get that K is a model of S_2^i. But we can prove more.

4.26 Theorem. Let $i \geq 0$, let T be S_2^1 if $i = 0$ and T_2^i otherwise. Let K be a $\Box_{i+1}^p$ closed substructure of some model M of T. Then K is a model of T_2^i.

Proof. Let $i \geq 0$, let $M \models T$, let $K \subseteq M$ be $\Box_{i+1}^p$ closed. We shall show that for every $\alpha(x, \mathbf{b})$ in Σ_i^b, a formula equivalent to $I(\alpha(x, \mathbf{b}))$ (induction for $\alpha(x, \mathbf{b})$) has $\Box_{i+1}^p$ Skolem functions, though this is not (known to be equivalent to) a $\Sigma_0^b(\Sigma_i^b)$ formula. Then, by Lemma 4.22, $I(\alpha(x, \mathbf{b}))$ must hold in K. We shall consider the following equivalent formula

$$\neg\alpha(\overline{0}, \mathbf{b}) \vee (\exists x < a)(\alpha(x, \mathbf{b}) \,\&\, \neg\alpha(x + \overline{1}, \mathbf{b})) \vee \alpha(a, \mathbf{b}) .$$

Again we shall use binary search. We start with $p := 0$ and $q := a$, and repeat:

$$p := \llcorner (p+q)/2 \lrcorner, \quad q := q, \quad \text{if } \alpha(\llcorner (p+q)/2 \lrcorner, \mathbf{b}),$$
$$p := p, \quad q := \llcorner (p+q)/2 \lrcorner, \quad \text{otherwise}.$$

If $\alpha(\overline{0}, \mathbf{b})$ and $\neg\alpha(a, \mathbf{b})$, then this algorithm finds in polynomially many steps (in the length of numbers $a, \mathbf{b}$) some c such that $c < a$, $\alpha(c, \mathbf{b})$, and $\neg\alpha(c+\overline{1}, \mathbf{b})$. It uses Σ_i^b oracle $\alpha(x)$. The existence of the computation follows from Theorem 4.17. We must show the correctness of the algorithm, which means that the value obtained by the computation witnesses the existential quantifier in the induction instance *provably in* T_2^i. We want to show that the following conditions are satisfied in the j-th step of computation:

(1) $\quad q - p \leq 2^{-j}$;

(2) $\quad \alpha(p, \mathbf{b})$;

(3) $\quad \neg\alpha(q, \mathbf{b})$.

We cannot prove by induction that (2) & (3) holds in each step, since this formula has too large complexity. So we prove (1) and (2) using $LIND\,\Sigma_i^b$ and then we prove separately (3) using $LIND\,\Pi_i^b$. □

Note that we are not able to prove the theorem with S_2^i instead of T_2^i, since we are not able to formalize $\Box_{i+1}^p$ functions in S_2^i. As a corollary we get a version of the theorem of Buss for fragments T_2^i. It will be a corollary of a stronger result which we shall prove later, but we have to prove it now, since we shall need it for the proof of the stronger one. Moreover the proof is much easier then the proof of Buss's theorem.

4.27 Theorem. Let $i \geq 0$, let $\varphi(x, y)$ be Π_i^b or $strict\hat{\Sigma}_{i+1}^b$ and suppose

$$T_2^i \vdash (\forall x)(\exists y)\varphi(x, y).$$

Then for some f in $\Box_{i+1}^p$

$$N \models (\forall x)\varphi(x, f(x)).$$

Moreover, if T is S_2^1 if $i = 0$ and T_2^i otherwise, then $T \vdash (\forall x)\varphi(x, f(x))$, i.e. more precisely, for some $e \in N$,

$$T \vdash (\forall x)(\exists y)(\{\overline{e}\}_{i+1}(x) = y \,\&\, \varphi(x, y)).$$

Proof. Since for every e, T proves that $\{\overline{e}\}_{i+1}(x) = y$ defines a function, we can use the functional notation. First suppose that φ is Π_i^b. By way of contradiction, suppose that T does not prove $(\forall x)\varphi(x, \{\overline{e}\}_{i+1}\{x\})$ for any e. Then also, for any k, T does not prove

$$\varphi(x, \{\overline{0}\}_{i+1}(x)) \vee \varphi(x, \{\overline{1}\}_{i+1}(x)) \vee \cdots \vee \varphi(x, \{\overline{k}\}_{i+1}(x)),$$

since otherwise, using $\varphi(x,y)$ as an oracle, we could combine these Turing machines into one producing y for a given x. By compactness, there is a model M for

$$T + \neg\varphi(c, \{\overline{0}\}_{i+1}(c)) + \neg\varphi(c, \{\overline{1}\}_{i+1}(c)) + \cdots ,$$

where c is a new constant. Let K be the closure of $\{c\}$ under $\Box^p_{i+1}$ functions in M, i.e

$$K = \{a \in M \mid (\exists e \in N)(M \vDash \{\overline{e}\}_{i+1}(c) = a)\} .$$

Then, for every $a \in K$, $M \vDash \neg\varphi(c,a)$. By Corollary 4.25, K is Σ^b_i elementary in M, hence $K \vDash (\forall y)\neg\varphi(c,y)$. By Theorem 4.26, $K \vDash T^i_2$, thus T^i_2 does not prove $(\forall x)(\exists y)\varphi(x,y)$.

Now suppose that $\varphi(x,y)$ is $strict\hat{\Sigma}^b_{i+1}$. Then for some $\hat{\Pi}^b_i$ formula $\psi(x,z_0,z_1)$ and some term t, $\varphi(x,y)$ is $(\exists z_1 \le t)\psi(x,y,z_1)$. Hence

$$(\forall x)(\exists y)\varphi(x,y) \equiv (\forall x)(\exists z)\psi(x,(z)_{\overline{0}},(z)_{\overline{1}}) \,\&\, (z_{\overline{1}} \le t) .$$

By the first part of the proof z can be witnessed by some $\{\overline{e}\}_{i+1}(x)$, hence y can be witnessed by $(\{\overline{e}\}_{i+1}(x))_{\overline{0}}$, which is a $\Box^p_{i+1}$ function. □

4.28 Corollary. Let $i \ge 1$, let $\Phi(x)$ be $strict\hat{\Delta}^b_{i+1}$ in T^i_2. Then T^i_2 proves induction for $\Phi(x)$.

Proof. The assumption of the theorem means that $\Phi(x)$ is $strict\hat{\Sigma}^b_{i+1}$ and for some $\Psi(x)$ $strict\hat{\Sigma}^b_{i+1}$, $T^i_2 \vdash \Phi(x) \equiv \neg\Psi(x)$. We can represent Φ and Ψ by

$$\Phi(x) \equiv (\exists y)\varphi(x,y) ,$$
$$\Psi(x) \equiv (\exists y)\psi(x,y) ,$$

where φ, ψ are in $\hat{\Pi}^b_i$. Thus we have

$$T^i_2 \vdash (\forall x)(\exists y)(\varphi(x,y) \vee \psi(x,y)) .$$

By Theorem 4.27 we get some e such that

$$T^i_2 \vdash \varphi(x, \{\overline{e}\}_{i+1}(x)) \vee \psi(x, \{\overline{e}\}_{i+1}(x)) ,$$

hence

$$T^i_2 \vdash \Phi(x) \equiv \varphi(x, \{\overline{e}\}_{i+1}(x)) .$$

We have not gained any reduction of complexity, but this was not our aim, our aim is to witness the induction formula for $\Phi(x)$ using a $\Box^p_{i+1}$ function. Suppose we have $\Phi(\overline{0})$ and $\neg\Phi(a)$, and we want to construct some $x < a$ such that $\Phi(x) \,\&\, \neg\Phi(x+1)$. We shall use binary search in a similar way as in

Theorem 4.26. We use $\varphi(x,y)$ as a Π_i^b oracle and $\{\overline{e}\}_{i+1}(x)$ as a subroutine in order to ask questions of the form $\varphi(x,\{\overline{e}\}_{i+1}(x))$. The computation goes as follows:

step 0: $p := 0, \quad q := a, \quad u := \{\overline{e}\}_{i+1}(0), \quad v := \{\overline{e}\}_{i+1}(a)$;

step j+1: $r := \{\overline{e}\}_{i+1}(\llcorner(p+q)/2\lrcorner)$;
if $\varphi(\llcorner(p+q)/2\lrcorner, r)$ then $p := \llcorner(p+q)/2\lrcorner, \quad u := r$
else $q := \llcorner(p+q)/2\lrcorner, \quad v := r$.

The correctness is proved by $LIND\hat{\Pi}_i^b$ showing that in the j-th step we have

(1) $$q - p \leq 2^{-j}a;$$

(2) $$\varphi(p,u)\,\&\,\psi(q,v).$$ □

A $\exists\Pi_i^b$ formula consists of a prefix of existential quantifiers followed by a Π_i^b formula.

4.29 Theorem. Let $i \geq 0$, let $\varphi(x,y,z)$ be a $\exists\Pi_i^b$ formula and suppose

$$T_2^i \vdash (\forall x)(\exists y)(\forall z)\varphi(x,y,z).$$

Then for some $f_0,\ldots,f_n$ in $\square_{i+1}^p$

(d.1) $$N \models (\forall x, z_0,\ldots,z_n)(\varphi(x,f_0(x),z_0) \vee \varphi(x,f_1(x,z_0),z_1) \vee \cdots \\ \cdots \vee \varphi(x,f_n(x,z_0,\ldots,z_{n-1}),z_n)).$$

Moreover, this is provable in S_2^1, if $i=0$, and in T_2^i, if $i>0$.

Proof. Let T be S_2^1, if $i=0$, and T_2^i, if $i>0$, let $\varphi(x,y,z)$ be a $\exists\Pi_i^b$ formula. Suppose T does not prove the formula in (d.1) for any choice of $f_0,\ldots,f_n$. Take some enumeration of functions in $\square_{i+1}^p$ such that

(1) the n-th function f_n depends on $\leq n$ arguments;
(2) each $f \in \square_{i+1}^p$ occurs in the enumeration infinitely many times.

(For instance, $\{e\}_{i+1}(((x_1,x_2),\ldots,x_e))$, where $(-,-)$ is the pairing function, has this property, assuming natural coding of Turing machines.) By compactness,

$$T + \neg\varphi(c,f_0(c),d_0) + \neg\varphi(c,f_1(c,d_0),d_1) + \cdots$$

where $c, d_0, d_1, \ldots$ are new constants, is a consistent theory. Let M be a model of this theory and let

$$K = \{f_0(c), f_1(c,d_0), f_2(c,d_0,d_1),\ldots\}.$$

Since all projections occur in the enumeration and each function occurs in the enumeration infinitely many times, we have

(3) $c, d_0, d_1, d_2, \ldots \in K$,
(4) K is $\Box^p_{i+1}$ closed.

By (3), we have

$$\forall a \in K\ \exists d \in K\ M \models \neg\varphi(c, a, d)\,.$$

Since $\neg\varphi$ is $\forall\Sigma^b_i$, using Corollary 4.25 we get

$$K \models (\forall y)(\exists z)\neg\varphi(c, y, z)\,. \qquad \square$$

Now we would like to strengthen Theorem 4.27 by taking the weaker assumption $S^{i+1}_2 \vdash (\forall x)(\exists y)\varphi(x, y)$ instead of $T^i_2 \vdash (\forall x)(\exists y)\varphi(x, y)$. The proof above does not work for S^{i+1}_2, since we are not able to show that a $\Box^p_{i+1}$ closed substructure is a model of S^{i+1}_2. It is worthwhile to realize the reason, then we shall better understand the forthcoming proofs. Let $(\exists y \le t)\psi(x, y)$ be a Σ^b_{i+1} formula, where ψ is Π^b_i. In a $\Box^p_{i+1}$ closed substructure the quantifier $\exists y$ can be witnessed by different elements of the form $\{\overline{e}\}_{i+1}(a)$. Since e runs over standard numbers, we can obtain a cut (for instance contained in some segment $[\overline{0}, b]$) for which $(\exists y \le t)\psi(x, y)$ holds, hence the induction fails. Therefore we shall take closures under some functions with *nonstandard indices*.

The basic idea of the proof is the same as above: if $(\exists y)\varphi(c, y)$ is not witnessed by a $\Box^p_{i+1}$ function, we construct a submodel in which it does not hold. Thus we must be careful when adding $\{e\}_{i+1}(c)$ for e nonstandard. We shall use overspill. If $(\exists y)\varphi(c, y)$ is not witnessed by any $\{e\}_{i+1}(c)$, for e standard, then this must be also true for all e up to some small nonstandard r_1. Thus we ensure the failure of $(\exists y)\varphi(c, y)$ by taking the closure only under functions with such small indices. However it is not so easy to ensure the induction.

Now we sketch the idea of the model-theoretic proof of this strengthening. It is essentially the proof of Wilkie with some changes. Let $i \ge 0$, let T be S^1_2 if $i = 0$ and T^i_2 otherwise. Suppose that for every $e \in N$, T does not prove $\varphi(x, \{\overline{e}\}_{i+1}(x))$. Take a model $M \models T$ with some $c \in M$, such that $M \models \neg\varphi(c, \{\overline{e}\}_{i+1}(c))$. By overspill this is true also for all $e < r_1$. We shall construct substructures consisting of some elements of the form $\{\overline{e}\}_{i+1}(c)$, for $e < r_1$, thus we shall have $\neg(\forall x)(\exists y)\varphi(x, y)$ in the substructures. So we have only to ensure $LIND\,\Sigma^b_{i+1}$. We shall do it in ω steps (we use only countable models) adding $LIND$ for one formula and one string of parameters at a time. Let us take the formula $\Psi(x) \equiv (\exists y \le t)\psi(x, y)$ considered above, and let a be already in our substructure K. We want to extend K so that it satisfies the following instance of $LIND\,\Sigma^b_{i+1}$:

$$\text{(d.2)} \qquad \neg\Psi(\overline{0}) \vee (\exists j \le |a|)(\Psi(j)\,\&\,\neg\Psi(j + \overline{1})) \vee \Psi(|a|)\,.$$

We take a suitable $r_2 \leq r_1$, and try to find a witness g_0 for $\exists y$ in $\Psi(0)$ of the form $\{e\}_{i+1}(c)$, $e \leq r_2$. If we succeed, we try to find a witness g_1 for $\Psi(1)$ of the form $\{e\}_{i+1}((c,(0,g_0)))$, $e \leq r_2$, and so on. The bound r_2 is chosen so small that we never construct $\{e\}_{i+1}(c)$ with $e \geq r_1$. Suppose, for example, that for some $j < |a|$, we have found g_j, but there is no witness g_{j+1} of the form $\{e\}_{i+1}((c,(j,g_j)))$ for $\Psi(j+\overline{1})$. Then we take the $\Box^p_{i+1}$ closure of $(c,(j,g_j))$ and K in M, and we have (d.2). We repeat this extension process for other formulae and other parameters. In order to preserve (d.2) we use a similar argument as we used for $\neg(\forall x)(\exists y)\varphi(x,y)$. In our particular case, $\Psi(j)$ is preserved by Σ^b_i elementary extensions, since it is $\exists\Pi^b_i$, while $\neg\Psi(j+\overline{1})$ will be preserved because we shall add only elements of the form $\{e\}_{i+1}(c)$, for $e < r_2$. The following key lemma formalizes one step of the above construction.

4.30 Lemma. Let $i \geq 0$, let T be S^1_2 if $i = 0$ and T^i_2 otherwise. Let M be a model of T, K substructure of M, $a, \mathbf{b} \in K$, let $\Phi(x,\mathbf{y})$ be $strict\hat{\Sigma}^b_{i+1}$ and let $\Psi(\mathbf{y})$ be $strict\hat{\Pi}^b_{i+1}$. Suppose that

(1) K is a $\Box^p_{i+1}$ closure of one element of M;
(2) K is not cofinal in M;
(3) $K \vDash \Psi(\mathbf{b})$.

Then there exists a substructure K^* such that $K \subseteq K^* \subseteq M$, (1)–(3) holds for K^* and

(4) $K^* \vDash \neg\Phi(\overline{0},\mathbf{b}) \vee (\exists j < |a|)(\Phi(j,\mathbf{b}) \,\&\, \neg\Phi(j+\overline{1},\mathbf{b})) \vee \Phi(|a|,\mathbf{b})$.

Proof. Let the assumptions be satisfied. Let K be generated by an element c. Since K is not cofinal in M, there is some $d \in M$ such that

$$x \in K \Rightarrow |x| < |d|\,.$$

Since every $x \in K$ is computed in polynomial time from c, and by overspill, we have

$$|c|^{r_0} \leq |d|\,, \tag{d.3}$$

for some r_0 nonstandard. Recall that

$$\{e\}_{i+1}(x) = y \equiv \nu_{i+1}(e,x,y,2^{|x|^e})\,,$$

where ν_{i+1} is Δ^b_{i+1}. By (d.3), we can write $\{e\}_{i+1}(x) = y$ as a Δ^b_{i+1} formula with an additional parameter d, since

$$\begin{aligned} M \vDash \nu_{i+1}(e,x,y,2^{|x|^e}) &\equiv (\forall z \leq d)(z = 2^{|x|^e} \rightarrow \nu_{i+1}(e,x,y,z)) \\ &\equiv (\exists z \leq d)(z = 2^{|x|^e} \,\&\, \nu_{i+1}(e,x,y,z))\,. \end{aligned}$$

Let

$$\Phi(x, \mathbf{b}) \equiv (\exists z \leq t(x))\varphi(x, z)\,,$$
$$\Psi(\mathbf{b}) \equiv (\forall z \leq s)\psi(z)\,,$$

where $\varphi(x, z)$ is Π_i^b and $\psi(z)$ is Σ_i^b; we shall omit the parameters $\mathbf{b}$ from now on. Consider the following formula

$$(\forall e < |r|)(\{e\}_{i+1}(c) \leq s \to \psi(\{e\}_{i+1}(x)))\,. \tag{d.4}$$

Since $K \models \Psi$, K is Σ_i^b elementary in M, and every element of K is of the form $\{e\}_{i+1}(c)$, for e standard, this formula is true in M, for every r standard. Also this formula is equivalent to a Δ_{i+1}^b formula in M. Since $S_2^1 \vdash I\Delta_1^b$, (Theorem 4.7) and for $i \geq 1$, $T_2^i \vdash I\Delta_{i+1}^b$ (Corollary 4.28), we can use overspill to deduce that (d.4) holds in M for some r nonstandard. Let

$$r_1 = \min(|r_0|, ||r||, |c|)\,. \tag{d.5}$$

Now we construct a Turing machine with an oracle which searches for the witness of the existential quantifier in condition (4) of the lemma. The machine works on input c as follows:

1: $j := -1$; $g := 0$;
2: *find the first* $e \leq r_1$ *such that*

$$\{e\}_{i+1}((c,(j,g))) \leq t(j+\overline{1}) \,\&\, \varphi(j+\overline{1}, \{e\}_{i+1}((c,(j,g))))\,;$$

3: *if such an* e *does not exist or* $j = |a| - 1$, *then print* (j, g);
4: *otherwise* $j := j + \overline{1}$; $g := \{e\}_{i+1}((c,(j,g)))$; *go to* 2 *where we define* $(-1, 0) = 0$.

Let e_0 be the code of this Turing machine. We shall estimate e_0. Recall that it is also the exponent in the time bound of this Turing machine and this makes this estimate nontrivial. To run some $\{e\}_{i+1}$ with $e \leq r_1$ on input c we need at most $|c|^e \leq |c|^{r_1}$ steps. In 2: we do it r_1 times. The time needed to check the Π_i^b condition of 2: is a standard polynomial in $|c|$. (Note that we have to evaluate also the implicit parameters $\mathbf{b}$, for which we need time a standard polynomial in $|c|$.) Finally the cycle given by 4: is repeated $|a| + 1$ times. Thus the total running time is bounded by

$$|c|^{r_1} * r_1 * p(|c|) * (|a| + 1)\,,$$

where $p(x)$ is some standard polynomial. We have $r_1 \leq |c|$ and $|a| + 1$ is also bounded by some standard polynomial in $|c|$, since it is an element of K. Thus we can bound the above expression by $|c|^{2r_1}$. Since the program of the Turing machine has standard length, we can bound its code by

$$e_0 \leq 2r_1\,.$$

Suppose the machine prints (j,g) on its output. We define K^* as follows. If $K \models \neg\Phi(\overline{0}) \vee \Phi(a)$, then $K = K^*$, otherwise K^* is the $\Box^p_{i+1}$ closure of

$$c^* = (c,(j,g)) = (c,\{e_0\}_{i+1}(c))\,.$$

Now we have to check conditions (2)–(4). We shall assume that $K \models \Phi(\overline{0}) \,\&\, \neg\Phi(a)$, hence $M \models \Phi(\overline{0})$, as K is Σ^b_i elementary in M.

We have

$$|c^*| \le |c|^{2r_1}\,,$$

hence the length of each element x of K^* is bounded by

$$|x| \le |c|^{k*2r_1}\,, \quad k \in N\,.$$

Since $r_1 \le |r_0|$, the last expression is bounded by

$$|c|^{r_0} \le |d|\,,$$

thus K^* cannot be cofinal in M.

To prove (3), suppose that it fails for K^*, which means that for some $e_1 \in N$ and $h = \{e_1\}_{i+1}(c^*)$,

$$K^* \models h \le s \,\&\, \neg\psi(h)\,.$$

By Corollary 4.25, K^* is Σ^b_i elementary in M, thus this formula holds also in M. Now c^* has been constructed from c using a machine with code $e_0 \le 2r_1$. Since e_1 is standard, the Turing machine which runs first e_0 and then e_1 has a code e_2 polynomial in r_1, hence, by (d.5), $e_2 \le |r|$. But this is in contradiction with (d.4).

It remains to prove condition (4) for K^*. We shall show that

$$K^* \models j < |a| \,\&\, \Phi(j) \,\&\, \neg\Phi(j+\overline{1})\,,$$

where (j,g) is the output of the machine e_0. The first inequality is clear, since the machine stops before reaching $|a|$. Also $\Phi(j)$ is easy, because, by the program lines 2: and 4:,

$$M \models g \le t(j) \,\&\, \varphi(j,g)\,,$$

hence also in K^* by elementariness. By way of contradiction suppose that $K \models \Phi(j+\overline{1})$. This means (again using elementariness) that, for some $e \in N$,

$$M \models \{e\}_{i+1}(c^*) \le t(j+\overline{1}) \,\&\, \varphi(j+\overline{1},\{e\}_{i+1}(c^*))\,.$$

In particular $e \le r_1$. Recall that $c^* = (c,(j,g))$, thus the machine should not stop at j, which is a contradiction. □

4.31 Theorem. Let $i \geq 0$, let T be S_2^1 if $i = 0$ and T_2^i otherwise. Let M be a *countable* model of T, K a substructure of M, $\mathbf{b} \in K$, and let $\Psi(\mathbf{y})$ be $strict\hat{\Pi}^b_{i+1}$. Suppose that

(1) K is a $\Box^p_{i+1}$ closure of one element of M;
(2) K is not cofinal in M;
(3) $K \models \Psi(\mathbf{b})$.

Then there exists a substructure K^* such that $K \subseteq K^* \subseteq M$ and

(4) $K^* \models \Psi(\mathbf{b})$;
(5) $K^* \models S_2^{i+1}$.

Proof. We shall construct a countable chain of substructures

$$K = K_0 \subseteq K_1 \subseteq K_2 \subseteq \cdots \subseteq M\,,$$

and $strict\hat{\Pi}^b_{i+1}$ formulae

$$\Psi = \Psi_0, \Psi_1, \Psi_2, \ldots\,.$$

Take some enumeration of $strict\hat{\Sigma}^b_{i+1}$ formulae with parameters from M. In the j-th step, $j > 0$, we ensure *LIND* in K_j for the first formula $\Phi(a, \mathbf{b})$ from this enumeration for which *LIND* fails in K_{j-1}, and whose parameters $\mathbf{b}$ are in K_{j-1}. Suppose we consider

$$\neg\Phi(\overline{0}, \mathbf{b}) \vee (\exists x < |a|)(\Phi(x, \mathbf{b}) \,\&\, \neg\Phi(x + \overline{1}, \mathbf{b})) \vee \Phi(|a|, \mathbf{b})$$

in the j-th step. We apply Lemma 4.30 to K_j, Φ and Ψ_j. Thus we obtain K_{j+1}. Then we define Ψ_{j+1} by

$$\begin{aligned}
&\Psi_{j+1} \equiv \Psi_j \,\&\, \neg\Phi(\overline{0}), \quad \text{if } K_{j+1} \models \neg\Phi(\overline{0})\,;\\
&\Psi_{j+1} \equiv \Psi_j \,\&\, \neg\Phi(k + \overline{1}),\\
&\qquad \text{if } K_{j+1} \models k < |a| \,\&\, \Phi(k) \,\&\, \neg\Phi(k+1)\,, \quad \text{for some } k \in K_{j+1}\,;\\
&\Psi_{j+1} \equiv \Psi_j, \quad \text{otherwise}.
\end{aligned}$$

Here Ψ_{j+1} is not written in the $strict\hat{\Pi}^b_{i+1}$ form; however to show that they are equivalent to such formulae, we need only properties of the coding relation that are expressible by equalities and hence preserved from M to $\Box^p_1$ closed substructures. Since Ψ_{j+1} will be true in all K_m, for $m \geq j+1$, also $LIND(\Phi(a, \mathbf{b}))$ will hold in these models. Consider for instance the second case, where $K_{j+1} \models k < |a| \,\&\, \Phi(k) \,\&\, \neg\Phi(k+1)$. Then $\Phi(k)$ will be preserved by extensions, since all the structures are $\Box^p_{i+1}$ closed, hence Σ^b_i elementary, while $\neg\Phi(k + \overline{1})$ will be preserved by the definition of the extensions. Let K^* be the union of this Σ^b_i elementary chain. Then all formulae Ψ_j are true in K^*, hence it satisfies condition (4) and

$$K^* \models LIND\text{-}strict\hat{\Sigma}^b_{i+1}\,.$$

By Lemma 4.23 this implies that K^* is a model of S_2^{i+1}. □

Now we are ready to prove Buss's witnessing theorem. The original theorem was proved with $T = S_2^{i+1}$; the next theorem is a strengthening (since T_2^i is a subtheory of S_2^{i+1}) which is also due to Buss.

4.32 Theorem. Let $i \geq 0$, let $\varphi(x, y)$ be $\hat{\Sigma}^b_{i+1}$. Suppose that

$$S_2^{i+1} \vdash (\forall x)(\exists y)\varphi(x, y).$$

Then for some f in $\Box^p_{i+1}$,

$$T \vdash (\forall x)\varphi(x, f(x)),$$

where T is S_2^1 if $i = 0$ and T_2^i otherwise.

Proof. First observe that we can bound y by some term t:

$$S_2^{i+1} \vdash (\forall x)(\exists y \leq t(x))\varphi(x, y).$$

This follows from Theorem 1.4. By Corollary 4.12 we can assume that φ is *strict*$\hat{\Sigma}^b_{i+1}$. Suppose that the conclusion is false. By compactness, there exists a countable model M of T such that

$$M \models \neg\varphi(c, \{e\}_{i+1}(c))$$

for every e standard and moreover, $2^{|c|^k}$, $k \in N$, is not cofinal in M. Let K be the $\Box^p_{i+1}$ closure of the element c in M. Then K is not cofinal in M and

$$K \models (\forall y)\neg\varphi(c, y),$$

since, by Corollary 4.25, it is Σ^b_i elementary in M. Let Ψ be $(\forall y \leq t)\neg\varphi(c, y)$, then it is equivalent to a *strict*$\hat{\Pi}^b_{i+1}$ formula already in predicate logic. Thus we can apply Theorem 4.31 to get some

$$K^* \models S_2^{i+1} + (\forall y \leq t)\neg\varphi(c, y),$$

which is a contradiction. □

The most interesting case is with $i = 0$; we state it explicitly.

4.33 Corollary. Let $\varphi(x, y)$ be a Σ^b_1 formula. Suppose that

$$S_2^1 \vdash (\forall x)(\exists y)\varphi(x, y).$$

Then for some f polynomial time computable function

$$S_2^1 \vdash (\forall x)\varphi(x, f(x)).$$ □

4.34 Corollary. For $i \geq 1$, S_2^{i+1} is a $\hat{\Sigma}_{i+1}^b$ conservative extension of T_2^i, i.e. for $\varphi(x)$ in $\hat{\Sigma}_{i+1}^b$,

$$S_2^{i+1} \vdash (\forall x)\varphi(x) \Rightarrow T_2^i \vdash (\forall x)\varphi(x).$$

Proof. Add a dummy existential quantifier to $\varphi(x)$ and apply Theorem 4.32. □

Recall that Δ_i^p sets are sets computable in polynomial time using oracles from Σ_{i-1}^p. Let us note that it is an open problem whether $\Sigma_{i+1}^p \cap \Pi_{i+1}^p = \Delta_i^p$. Thus Δ_i^b definable sets need not be in Δ_i^p.

4.35 Corollary. Let $i \geq 1$, let $\varphi(x)$ be a Δ_i^b formula with respect to S_2^i. Then $\varphi(x)$ defines a Δ_i^p subset of N.

Proof. For $\varphi(x)$ we can find some formulae $\alpha(x,y), \beta(x,y)$ in $\hat{\Pi}_{i-1}^b$ and terms t, s such that

$$S_2^i \vdash \varphi(x) \equiv (\exists y \leq t)\alpha(x,y) \equiv \neg(\exists y \leq s)\beta(x,y).$$

Hence

$$S_2^i \vdash (\exists y \leq t)\alpha(x,y) \vee (\exists y \leq s)\beta(x,y).$$

The last formula can be easily transformed into a $\hat{\Sigma}_i^b$ formula; thus we can apply Theorem 4.32 to obtain some f in $\square_i^p$ such that

$$N \models (\forall x)((f(x) \leq t \,\&\, \alpha(x, f(x))) \vee (f(x) \leq s \,\&\, \beta(x, f(x)))).$$

Hence we can decide $\varphi(x)$ by computing $f(x)$ and using a Π_{i-1}^p oracle. □

Again the most interesting particular case of the last corollary is the one with $i = 0$; it can be stated as follows

$$S_2^1 \vdash X \in NP \cap coNP \Rightarrow S_2^1 \vdash X \in P.$$

(e) On the Finite Axiomatizability of Bounded Arithmetic

In this subsection we shall consider the question of the finite axiomatizability of Bounded Arithmetic. First we shall show that the fragments S_2^i and T_2^i are finitely axiomatizable for $i \geq 1$. Thus the finite axiomatizability of S_2 is equivalent to the statement that the hierarchy of these theories is infinite.

Then we shall reduce the finite axiomatizability of S_2 to the non-collapsedness of Polynomial Hierarchy.

4.36 Theorem. For $i \geq 1$, each of S_2^i and T_2^i is finitely axiomatizable.

Proof-sketch. The basic idea of the proof is the same as for fragments $I\Sigma_n$, see Theorem I.2.52. However, since this theorem is important, we give at least a sketch of the proof.

In Theorem 4.18 we showed that there are Σ_i^b formulae μ_i which are in a sense universal; more precisely, for every $\varphi(x)$ in Σ_i^b there exists an e such that

$$S_2^1 \vdash \varphi(x) \equiv \mu_i(\overline{e}, x, 2^{|x|^{\overline{e}}}).$$

Furthermore, $2^{|x|^{\overline{e}}}$ provably exists in S_2^1; this follows from the bound

$$2^{|x|^{y+1}} \leq 2^{|x|^y} \# x.$$

For induction and *PIND* we need formulae which have an induction variable and parameters. One parameter is sufficient since the pairing function and the decoding functions are Δ_1^b definable. For the same reason, we can extend the result above to formulae with two free variables:

$$S_2^1 \vdash \varphi(x,y) \equiv \mu_i(\overline{e}, (x,y), 2^{|x|^{\overline{e}}}).$$

Another easy modification of the result above reads as follows:

$$S_2^1 \vdash z \geq 2^{|(x,y)|^{\overline{e}}} \rightarrow (\varphi(x,y) \equiv \mu_i(\overline{e}, (x,y), z)).$$

Now we only need to show that there exists a finite subtheory T of S_2^1 such that *for all* $\varphi(x,y)$ in Σ_i^b the above equivalence is provable in T. Once we have such a T, we can finitely axiomatize S_2^i by T plus *BASIC* plus

$$(\forall e, y, z) PIND(z \geq 2^{|(x,y)|^e} \rightarrow \mu_i(e, (x,y), z)),$$

where x is the induction variable (thus bounded in *PIND*), and e, y, z are parameters of induction. Similarly we can axiomatize T_2^i.

To find such a T we have to analyze formulae μ_i and ν_i of Theorem 4.18 more closely. The concept that we want to use is just Tarski's conditions for the definition of the satisfaction of Σ_i^b formulae. Tarski's conditions describe the satisfaction of formulae of a given class via relating the truth of a formula with the truth of its components. For each closure condition of the class of formulae there is one Tarski condition. The classes Σ_i^b were defined in Definition 4.2; furthermore we have to use the inductive conditions for the value of terms.

Now we shall describe the conditions more explicitly.

(1) For each function symbol of L_2 we shall have an integer constant, e_S, e_+, e_*, etc., such that

$$S_2^1 \vdash \nu_0(\overline{e}_S, x, y, 2^{|x|^{\overline{e}_S}}) \equiv y = S(x);$$
$$S_2^1 \vdash \nu_0(\overline{e}_+, (x_1, x_2), y, 2^{|(x_1,x_2)|^{\overline{e}_+}}) \equiv y = x_1 + x_2;$$

etc.

(2) For relation symbols $=$ and $\leq$ we construct binary functions $e_=$ and $e_\leq$ such that

$$S_2^1 \vdash \mu_0(e_=(e_1, e_2), x, 2^{|x|^{e_=(e_1,e_2)}})$$
$$\equiv (\exists y_1, y_2)(\nu_0(e_1, x, y_1, 2^{|x|^{e_1}}) \,\&\, \nu_0(e_2, x, y_2, 2^{|x|^{e_2}}) \,\&\, y_1 = y_2);$$

and similarly for $\leq$. The functions are Δ_1^b.

(3) For each of the conditions of the definition of the class Σ_i^b we have a function and a condition. Consider for instance the condition

$$\alpha \in \Sigma_i^b \Rightarrow (\forall z \leq |t|)\alpha \in \Sigma_i^b.$$

Then we have a binary Δ_1^b definable function $e_{|\forall|}$ such that

$$S_2^1 \vdash \mu_i(e_{|\forall|}(e_1, e_2), x, 2^{|x|^{e_{|\forall|}}})$$
$$\equiv (\forall y, z)(\nu_0(e_1, x, y, 2^{|x|^{e_1}}) \,\&\, z \leq |y| \to \mu_i(e_2, (x, z), 2^{|(x,z)|^{e_2}})).$$

The function computed by the machine with the code $e_{|\forall|}$ does the following. If e_1 is the code of a machine for a term t and e_2 is the code of a machine for a formula $\varphi(x)$, then $e_{|\forall|}(e_1, e_2)$ is the code of a machine which decides the formula $(\forall z \leq |t|)\varphi(x, z)$.

Let T consist of the above conditions plus finitely many sentences needed to formalize functions $e_=$, $e_\leq$, $e_{|\forall|}$, ... (existence, uniqueness, Δ_1^b definition). Then we can easily show, using induction on the depth of $\varphi \in \Sigma_i^b$, that there exists an e such that

$$T \vdash \varphi(x) \equiv \mu_i(\overline{e}, x, 2^{|x|^{\overline{e}}}).$$

It remains to prove the above Tarski conditions in S_2^1. A formal proof would, perhaps, require a whole section. But in fact it is not hard to see that they must be provable. Let us look at the last considered condition, which concerns the sharply bounded universal quantifier. Formula μ_i has been defined as being true if the Turing machine with code e accepts input x. The condition describes what the machine with code e of the form $e_{|\forall|}(e_1, e_2)$ will do: first it will simulate machine e_1 on x and obtain y (the value of the

term), then it will simulate machine e_2 on inputs (x, z) with $z \leq |y|$. Thus we need only to choose the codes of the machines in a suitable way. □

4.37 Corollary. S_2 is finitely axiomatizable if and only if $S_2 = S_2^i$ for some i.

We are not able to decide whether S_2 is finitely axiomatizable. We shall only show that it is not, *assuming a plausible conjecture* that the Polynomial Hierarchy does not collapse on its finite level. This does not seem to be a surprising result because of the relation between the classes of formulae Σ_i^b and complexity classes Σ_i^p. But the relation is not so straightforward. Induction and definability are different things. We shall show that induction is related to optimization and we shall use this fact in our proof.

4.38 Theorem. Let $i \geq 0$ and suppose $\Sigma_{i+2}^p \neq \Pi_{i+2}^p$. Then $T_2^i \neq S_2^{i+1}$.

4.39 Corollary. If the Polynomial Hierarchy does not collapse then
(i) $S_2^i \neq S_2^{i+1}$ for $i \geq 1$ and $T_2^i \neq T_2^{i+1}$ for $i \geq 0$.
(ii) S_2 and T_2 are not finitely axiomatizable.
(iii) $I\Delta_0 + \Omega_1$ is not finitely axiomatizable.
(iv) $I\Delta_0$ is not finitely axiomatizable.

Proof. (i) and (ii) follow using the inclusions $T_2^0 \subseteq S_2^1 \subseteq T_2^1 \subseteq S_2^2 \cdots$.

(iii) $I\Delta_0 + \Omega_1$ is contained in S_2 in such a way that the functions of S_2 are definable in $I\Delta_0 + \Omega_1$.

(iv) Just recall that Ω_1 is a single axiom. □

Before proving Theorem 4.38 formally we shall sketch the main steps of this proof. We shall consider only the case $i = 0$.

Let $C(x, y)$ be a polynomial time predicate. Consider the optimization problem of finding maximal y such that $y \leq x$ and $C(x, y)$. In order to simplify notation we shall assume that the bound $y \leq x$ is implicit in $C(x, y)$ and that $C(x, 0)$ holds for all x. We shall call y *a feasible solution to* x if $C(x, y)$. Let φ be defined by

$$\varphi(x, y, z) \equiv C(x, y) \,\&\, (|y| < |z| \rightarrow \neg C(x, z)) . \tag{e.1}$$

Since C is polynomial time computable, φ is Σ_1^b. Now the meaning of $(\forall z)\varphi(x, y, z)$ is that y *is the feasible solution to* x *of maximal size.* In S_2^1 the schema $LIND\Sigma_1^b$ proves that for every x there exists a maximal feasible solution y. If $T_2^0 = S_2^1$, then this is true also in T_2^0, i.e.

$$T_2^0 \vdash (\forall x)(\exists y)(\forall z)\varphi(x, y, z) .$$

Now we can apply Theorem 4.29. We obtain some functions $f_0, f_1, \ldots, f_n$ computable in polynomial time such that

$$(e.2) \quad N \models (\forall x, z_0, \ldots, z_n)(\varphi(x, f_0(x), z_0) \vee \varphi(x, f_1(x, z_0), z_1) \vee \cdots \cdots \vee \varphi(x, f_n(x, z_0, \ldots, z_{n-1}), z_n)) .$$

The rest of the proof is only complexity theory. We want to show that such functions cannot exist for all polynomial time predicates C. If n were always 0 (as in Buss's theorem) then our goal would be simple. In such a case, for example, we could construct a maximal clique in a graph using a polynomial time computable function f_0. This is impossible unless $P = NP$. In fact we can take any NP problem, not only optimization problems. Let $C(x, y)$ be defined, say, by

$$y = 0 \text{ or } y \text{ is a Hamiltonian circuit in graph } x .$$

Then $f_0(x)$ would be 0, if x is not Hamiltonian and $f_0(x)$ would be a Hamiltonian circuit in x otherwise.

But in general n need not be 0. So we modify our problem as follows: x will be a string of graphs and y will be a string of 0's and Hamiltonian circuits in corresponding graphs. Now we assume that for this particular C we obtain (e.2) with $n = 1$, (this is sufficiently instructive). Consider an x, a two element string $x = (G_1, G_2)$, where both graphs are Hamiltonian. Then there are two possibilities:

(1) $f_0(x)$ is not a feasible solution or $f_0(x) = (y_1, y_2)$ where $y_1 = y_2 = 0$;
(2) $f_0(x) = (y_1, y_2)$ and y_1 is a Hamiltonian cycle in G_1 or y is a Hamilton cycle in G_2;

In case (2) f_0 has produced nontrivial information. In case (1) it is not so, but f_1 must produce information: if we take $z_0 = (y', 0)$, where y' is some Hamiltonian cycle in G_1, then we have $\neg\varphi(x, f_0(x), z_0)$, since either $f_0(x)$ is not a feasible solution or z_0 is a better one. Hence we get

$$(\forall z_1)\varphi(x, f_1(x, z_0), z_1) ,$$

i.e. $f_1(x, z_0)$ *is* an optimal solution. This means that $f_1(x, z_0) = (y_1, y_2)$ where both y_1 and y_2 are Hamilton cycles. Let us draw an arrow $G_1 \to G_2$ in case (1) and an arrow $G_2 \to G_1$ in case (2). This indicates that by having a Hamiltonian circuit for the tail we can construct a Hamiltonian circuit for the head. Consider all Hamiltonian graphs of size m. The above argument shows that there is an arrow at least in one direction between any pair of them. An easy counting argument shows that there exists a set of polynomial size of such graphs which covers the rest. For each of these graphs, we choose a Hamiltonian circuit in it. Let $\mathbb{H}$ be the set consisting of these pairs. $\mathbb{H}$ has also polynomial size. Using $\mathbb{H}$ we can decide in polynomial time whether a graph G is Hamiltonian: try f_0 and f_1 on all pairs (G, H) and (H, G), where H runs through $\mathbb{H}$. Computations which use additional polynomial size

information are called computations with *polynomial advice*. There is a well-known argument (see [Karp-Lipton 80]) showing that if every *NP* problem can be computed using polynomial time computation with polynomial advice, then $\Sigma_2^p = \Pi_2^p$.

Now we prove Theorem 4.38 in detail. We shall divide the proof into several lemmas. Let $i \geq 0$ be given. A formula $C(x, y)$ will be called a Π_i^b *optimization problem* if $C(x, y)$ is a Π_i^b formula such that $C(x, \overline{0})$ and $C(x, y) \to y \leq x$ is provable in predicate logic. We shall say that y is a *length optimal* solution to x if the following formula holds:

$$(\forall z)(C(x, y) \,\&\, (|y| < |z| \to \neg C(x, z))) .$$

4.40 Proposition. For every Π_i^b optimization problem C, S_2^{i+1} proves that C has a length optimal solution to every x.

Proof. Let $\psi(x, y)$ be defined by

$$\psi(x, u) \equiv (\exists y \leq x)(C(x, y) \,\&\, |y| \geq u) .$$

By $LIND\,\Sigma_{i+1}^b$ we have

$$\neg\psi(x, \overline{0}) \vee (\exists u < |x|)(\psi(x, u) \,\&\, \neg\psi(x, S(u))) \vee \psi(x, |x|) .$$

Since $\psi(x, \overline{0})$ is always true, it just expresses the existence of a length optimal solution. □

It is an easy exercise to prove that, in fact, the existence of length optimal solutions to Π_i^b optimization problems is *equivalent to* $PIND\,\Sigma_{i+1}^b$ over a sufficiently strong base theory, say S_2^1.

We are going to use Theorem 4.29. It is convenient to refer to the conclusion of the theorem, see formula (e.2) above, as a kind of interactive way of computing a witness to $\exists y$ in $(\forall x)(\exists y)(\forall z)\varphi(x, y, z)$. Let x be given and suppose we want to construct y. Assume (e.2) holds true. First consider $f_0(x)$. If $(\forall z)\varphi(x, f_0(x), z)$, then take $y = f_0(x)$, otherwise there is some z_0 such that $\neg\varphi(x, f_0(x), z_0)$. We shall call this z_0 a *counterexample* to $f_0(x)$. By (e.2), it must be $n > 0$ and we have

$$N \vDash (\forall x, z_1, \ldots, z_n)(\varphi(x, f_1(x, z_0), z_1) \vee \cdots \\ \cdots \vee \varphi(x, f_n(x, z_0, \ldots, z_{n-1}), z_n)) .$$

Hence we can repeat the reasoning above and we get that either $f_1(x, z_0)$ witnesses y or the disjunction can be reduced further by taking a counterexample z_1 to $f_1(x, z_0)$. However this process can be repeated at most n times, since when we obtain

$$(\forall x, z_n)\varphi(x, f_n(x, z_0, \ldots, z_{n-1}), z_n)$$

we can stop.

Let $C(x,y)$ be a Π_i^b optimization problem. Assume that $T_2^i = S_2^{i+1}$. Then T_2^i proves that C has always length optimal solutions. This statement can be expressed in the form $(\forall x)(\exists y)(\forall z)\varphi(x,y,z)$ with φ in Σ_{i+1}^b, see (e.1). Hence, by Theorem 4.29, we get that an optimal solution can be computed using $\square_{i+1}^p$ functions and a fixed number of counterexamples. This is our first lemma.

4.41 Lemma. If $T_2^i = S_2^{i+1}$, $i \geq 0$, then for every Π_i^b optimization problem its length optimal solutions can be computed using $\square_{i+1}^p$ functions and a fixed number of counterexamples.

The computations with counterexamples do not define a function complexity class, since we allow the asking of counterexamples only for the particular optimization problem that we want to solve. If we allowed asking arbitrary queries of the same logical complexity, we would obtain just the class $\square_{i+2}^p$. But it is easy to prove that optimal solutions can be computed by such functions. Thus our next step is a reduction to a different function class.

4.42 Definition. A function f belongs to $\square_i^p/poly$ if there exists some g in $\square_i^p$ and a polynomial $p(x)$ such that for every k there is $a_k \leq p(k)$ such that for every x, $|x| = k$,

$$f(x) = g(x, a_k).$$

Here a_k is called a *polynomial advice*; it is some extra information given for free for each size of input; the dependence on the size of input is quite arbitrary.

4.43 Lemma. Let $i \geq 0$. Suppose that for every Π_i^b optimization problem its length optimal solutions can be computed using $\square_{i+1}^p$ functions and a fixed number of counterexamples. Then for every $\psi(x,y)$ in Π_i^p there exists an f in $\square_{i+1}^p/poly$ such that

$$N \models (\forall x)((\exists y \leq x)\psi(x,y) \rightarrow \psi(x, f(x))).$$

Proof. Let ψ be given. Define a Π_i^p optimization problem $\rho(u,v)$ by

$$\rho(u,v) \equiv lh(v) \leq lh(u) \,\&\, (\forall t < lh(v))(\psi((u)_t,(v)_t) \,\&\, (v)_t \leq (u)_t).$$

Let $f_0(u), \ldots, f_n(u, v_0, \ldots, v_{n-1})$ be some functions in $\square_{i+1}^p$ which interactively compute length optimal solutions for ρ. Let k be given. We shall construct a polynomial advice for inputs of size k. Once we describe the advice, the definition of the function g will be clear. Let

$$V_1 = \{x; |x| = k \,\&\, (\exists y \leq x)\psi(x,y)\}.$$

Choose some function $w(x)$ such that, for $x \in V_1$,

$$\psi(x, w(x)) .$$

We shall use the following notation. If $m < lh(u)$ then

$$w(u \mid m)$$

denotes the sequence $(w((u)_0), \ldots, w((u)_m))$. To each $n+1$-tuple u of elements of V_1, we assign a pair (l, y), $0 \leq l \leq n$, using the following procedure:

Step 0: compute $f_0(u)$; if $\rho(u, f_0(u))$ and $lh(f_0(u)) > 0$, then put $l = 0$ and $y = (f_0(u))_0$ and stop, otherwise go to step 1;

...

Step m ($m < n$): compute $f_m(u, w(u \mid 0), \ldots, w(u \mid m-1))$; if

$$\rho(u, f_m(u, w(u \mid 0), \ldots, w(u \mid m-1)))$$

and

$$lh(f_m(u, w(u \mid 0), \ldots, w(u \mid m-1))) > m ,$$

then put $l = m$ and

$$y = (f_m(u, w(u \mid 0), \ldots, w(u \mid m-1)))_m ,$$

otherwise go to step $m+1$;

...

Step n: If we have reached this step, then it necessarily holds that

$$\rho(u, f_n(u, w(u \mid 0), \ldots, w(u \mid n-1)))$$

and

$$lh(f_n(u, w(u \mid 0), \ldots, w(u \mid n-1))) = n+1 ,$$

thus we put $l = n$ and

$$y = f_n(u, w(u \mid 0), \ldots, w(u \mid n-1)) ,$$

and stop.

Let us call y a *witness* for x if $\psi(x, y)$. The meaning of the above procedure is the following. Let u be an $n+1$-tuple and let l, y be assigned to it. Then having witnesses for $(u)_0, \ldots, (u)_{l-1}$, we can compute the witness y for $(u)_l$.

For an n-element subset Q of V_1 and $x \in V_1 \backslash Q$, we shall say that pair (Q, x) is *good* if for some arrangement $\{x_0, \ldots, x_{l-1}, x_{l+1}, \ldots, x_n\}$ of Q, l is assigned to u by the above procedure, where u is the sequence

$$(x_0, \ldots, x_{l-1}, x, x_{l+1}, \ldots, x_n) .$$

Define a sequence of subsets of V_1

$$V_1 \supseteq V_2 \supseteq V_3 \cdots ,$$

having $N_1, N_2, \ldots$, respectively, elements. The $j+1$-st element in the sequence is defined as follows. Find an n-element subset $Q_j \subseteq V_j$ such that

$$|\{x \in V_j; (Q_j, x) \textit{ is good }\}| \geq \frac{N_j - n}{n+1},$$

and take

$$V_{j+1} := V_j \backslash \{x \in V_j; (Q_j, x) \textit{ is good }\}.$$

We must show that it is always possible to choose such a Q_j. By the procedure above, for each $n+1$-element subset $\{x_0, \ldots, x_n\}$ of V_j we can construct a good pair (Q, x) such that

$$\{x_0, \ldots, x_n\} = Q \cup \{x\},$$

by taking u to be the sequence $(x_0, \ldots, x_n)$. Hence there are at least $\binom{N_j}{n+1}$ good pairs. On the other hand there are $\binom{N_j}{n}$ n-element subsets Q of V_j, so at least one such Q must form good pairs with at least

$$\binom{N_j}{n+1} * \binom{N_j}{n}^{-1} = \frac{N_j - n}{n+1}$$

elements. Hence

$$N_{j+1} \leq N_j - \frac{N_j - n}{n+1} = \frac{n}{n+1}(N_j + 1),$$

from which we get

$$\begin{aligned} N_{j+1} &\leq \left(\frac{n}{n+1}\right)^j N_1 + \frac{n}{n+1} + \left(\frac{n}{n+1}\right)^2 + \left(\frac{n}{n+1}\right)^3 + \cdots \\ &< \left(\frac{n}{n+1}\right)^j N_1 + n + 1. \end{aligned}$$

Hence we get $N_t \leq n+2$ after t steps, for

$$t \leq \log_2(N_1).(\log_2((n+1)/n))^{-1} \leq O(\log_2(2^k)) = O(k).$$

We take the polynomial size advice to be the sequence of pairs $(x, w(x))$, where x runs trough all elements of

$$Q_1 \cup Q_2 \cup \cdots \cup Q_t \cup V_t.$$

Now let x be such that

$$|x| = k \,\&\, (\exists y \leq x)\psi(x, y)\,,$$

i.e. $x \in V_1$. Then either $x \in V_t$ and so a witness for x is in the advice, or (Q_j, x) is a good pair for some $j \leq t$. But then we can construct a witness for x from the witnesses for the elements of Q_j (using $\square^p_{i+1}$ functions in the procedure above). □

4.44 Lemma. Let $i \geq 0$. Suppose that for every $\psi(x, y)$ in Π^p_i there exists an f in $\square^p_{i+1}/poly$ such that

$$N \models (\forall x)((\exists y \leq x)\psi(x, y) \to \psi(x, f(x)))\,.$$

Then $\Sigma^p_{i+2} = \Pi^p_{i+2}$.

Proof. Let $A(b)$ be in Π^p_{i+2}. Without loss of generality we can assume that it can be represented in the form

$$(\forall x)(|x| = |b| \to (\exists y \leq b)\gamma(b, x, y))\,,$$

where γ is Π^b_i.

Let $\psi(x, y)$ be a Π^b_i formula such that

$$\psi((b, x), y) \equiv |x| = |b| \to (y \leq b \,\&\, \gamma(b, x, y))\,.$$

Let f be a function in $\square^p_{i+1}/poly$ guaranteed by Lemma 4.43 for ψ, and let function g in $\square^p_{i+1}$ and polynomial p be guaranteed by Definition 4.42 for f. Then we can write $A(b)$ in the following Σ^b_{i+2} form[1]

$$A(b) \equiv (\exists a)(a \leq p(|(b, b)|) \,\&\, (\forall x)(|x| = |b| \to \gamma(b, x, g(a, (b, x))))))\,.$$

The right to left implication is trivial, the left to right implication follows by taking an advice a. □

This finishes the proof of Theorem 4.38.

There is a similar reduction of the question $S^i_2 = T^i_2$? to a problem in complexity theory. First we shall define that for an optimization problem $C(x, y)$, y is an *optimal* solution to x if

$$(\forall z)(C(x, y) \,\&\, (y < z \to \neg C(x, z)))\,.$$

Secondly, we define interactive computations with an unbounded number of counterexamples as the the natural extension of the concept above. Of course,

[1] Note that g is Π^b_{i+1} definable by Theorem 4.17.

there is an implicit polynomial bound to the number of counterexamples, since we consider only polynomial time oracle computations. Note that the number of feasible solutions to x may be exponential in $|x|$, thus polynomial number of counterexamples apparently cannot help to solve some difficult optimization problems (such as the TRAVELLING SALESPERSON), while it gives trivial algorithms if we ask only for *length* optimal solutions (as in CLIQUE for instance).

We have the following counterpart of Theorem 4.29.

4.45 Theorem. Let $i \geq 0$, let $S_2^i \vdash (\forall x)(\exists y)(\forall z \leq t)\varphi(x,y,z)$, for φ in Σ_{i+1}^b and t a term. Then, for a given x we can compute y using $\square_{i+1}^p$ computations with (an unbounded number of) counterexamples.

This theorem can be proved using the proof theoretical method which Buss used for his witnessing theorem. We are not going to prove it here. The counterpart of Lemma 4.41 is the following.

4.46 Corollary. For $i \geq 1$, if $S_2^i = T_2^i$, then the optimal solutions for every Π_{i-1}^b problem can be computed using $\square_{i+1}^p$ computations with (an unbounded number of) counterexamples.

We conjecture that the conclusion of Corollary 4.46 is false, hence also that $S_2^i \neq T_2^i$.

*

The research on witnessing functions is still going on and we can look for new nice results. The task of proving that Bounded Arithmetic is not finitely axiomatized *without assumptions on complexity classes* seems still too hard. What might be more accessible is to prove nonconservation results for S_2^i and T_2^i using such assumptions.

5. Interpretability and Consistency

(a) Introduction

In the last section we were mainly interested in the strength of theories obtained by restricting the quantifier complexity of bounded formulae in the scheme of induction. The main theme of this section is investigation of the strength of theories obtained from $I\Sigma_0$ by adding functions of a different growth rate. (These functions have graphs definable by Σ_0 formulae, so the strength is increased by assuming that they are total.) We shall

consider typical questions of metamathematics: interpretations, provability of consistency, conservativity, etc.

First we shall consider the definition of truth for Σ_0 formulae in Bounded Arithmetic. As a consequence we obtain that $I\Sigma_0 + Exp$ is finitely axiomatizable. Another consequence (which uses further technical results) is the existence of an interpretation of $I\Sigma_0 + \Omega_n$ in Q, $n = 1, 2, \ldots$, which in turn implies that Q and these theories are equiconsistent in $I\Sigma_0 + \Omega_1$. For further results we need a formalization of a weaker form of the cut-elimination theorem in $I\Sigma_0 + Exp$. The usual form of this theorem is unprovable in this theory. On the other hand Gödel's incompleteness theorem can be strengthened further for such fragments. For instance $I\Sigma_0 + Exp$ does not prove the consistency of Q. This and several other results of this kind will be shown in subsection (f). Finally we shall consider the question of limited use of exponentiation in $I\Sigma_0$.

(b) Truth Definitions for Bounded Formulae

In Chap. I, Sect. 1 we have shown that truth for Σ_0 formulae can be defined by a Δ_1 formula in $I\Sigma_1$. Because of diagonalization there is no such Σ_0 definition. However we can omit the main $\exists$ quantifier in the Σ_1 formula and thus we obtain a Σ_0 formula with an additional parameter. Then we estimate how large the parameter must be; put another way, we are looking for a suitable bound to the $\exists$ quantifier in the Σ_1 definition of Σ_0 truth. Sometimes we shall call the Σ_0 formula obtained in this way *a truth definition* for Σ_0 formulae, though it is not quite precise. The construction will be essentially the same as in Chap. I, only we have to compute the bounds more precisely.

Before constructing a formula with a parameter which defines the truth for Σ_0 formulae, we shall briefly consider this problem from the point of view of complexity theory. Assume that there is a definition of Σ_0 truth described above; suppose that it has the quantifier complexity k and the bound to the parameter is some function f. Then we have the following relation between complexity classes $\Sigma_0^N \subseteq \Sigma_k^{|f|}$ (see Sect. 2 for definitions). Thus the corresponding question is: for which function g and constant k do we have $\Sigma_0^N \subseteq \Sigma_k^g$? Recall that Σ_0^N are just the sets in the Linear Time Hierarchy (Theorem 2.16), hence the best that we can achieve (to our present knowledge) is

$$\Sigma_0^N \subseteq \bigcup_{c \in N} Time(c^n) = E\,.$$

This result can be used to write a definition of truth for Σ_0 formulae based on a Σ_0 formula with an exponential bound to the *length* of the parameter, but again we not only need a formula, but also proofs of its properties in $I\Sigma_0$. Furthermore we need an estimate of the constant c. Therefore we present an explicit construction.

We shall use the notation introduced in Sect. 3. Thus for instance $\max(s)$ is the maximal element of sequence s. In this subsection we use only the standard arithmetical language L_0. The formalization of syntax is as described in Sect. 3; thus if x is the Gödel number of an expression, $|x|$ is its length.

Our approach is to consider a bounded formula literally as a program. The computation of terms needs no explanation; the interpretation of bounded quantifiers is as search procedures. The crucial thing is to estimate the size of the numbers that will occur in such a computation.

Let us consider an example first. Let $\alpha(x)$ be a bounded formula

$$(\exists y \leq x^2)(\exists z \leq y^2)(\exists u \leq z^2)\beta(x, y, z, u)\,,$$

where β is also a bounded formula. To compute the truth value of $\alpha(x)$ we use three *FOR* loops, one for each bounded variable. Note that the third loop has the range 0 to x^{2^3}.

5.1 Proposition ($I\Sigma_0$). (1) For a term t and a string of numbers $\mathbf{s}$,

$$t(\mathbf{s}) \leq (\max(\mathbf{s}) + 2)^{|t|}\,.$$

(2) The maximal value needed to compute the truth of a bounded formula α on a string $\mathbf{s}$ is bounded from above by

$$(\max(\mathbf{s}) + 2)^{2^{|\alpha|}}\,.$$

Note in passing that, by (1), the value of a *closed* term t is bounded by $2^{|t|} \leq 2t$, hence it can be defined as a total function in $I\Sigma_0$, while for general terms in L_0 we need $I\Sigma_0 + \Omega_1$. Nevertheless the estimates above are provable in $I\Sigma_0$ provided that the upper bound exists.

Proof. (1) is easily proved by induction on the length of t.

(2) This is also proved by induction on the length of α. We shall compute only the induction step for a quantifier. Let $\alpha(\mathbf{s})$ be $(\exists y \leq t(\mathbf{s}))\beta(y, \mathbf{s})$ (for $\forall$ it is the same). Assume we have bound

$$(\max(\mathbf{s}, y) + 2)^{2^{|\beta|}}\,,$$

for $\beta(y, \mathbf{s})$. By (1) we know that

$$t(\mathbf{s}) \leq (\max(\mathbf{s}) + 2)^{|t|}\,,$$

hence for $\alpha(\mathbf{s})$ we get

$$\begin{aligned}(\max(\mathbf{s}, (\max(\mathbf{s}) + 2)^{|t|}) + 2)^{2^{|\beta|}} &= (((\max(\mathbf{s}) + 2)^{|t|}) + 2)^{2^{|\beta|}} \\ &\leq ((\max(\mathbf{s}) + 2)^{|t|+1})^{2^{|\beta|}}\,.\end{aligned}$$

We shall estimate the exponent:

$$(|t|+1)2^{|\beta|} \leq 2^{|t|+1+|\beta|} \leq 2^{|\alpha|},$$

which proves the induction step. □

First we define the graph of the *val* function which defines the value of arithmetical terms. Also here we need an additional parameter. The meaning of $Value(y,x,z,u)$ is: y is the value of term x computed on string z, u is the parameter. Let $Var(x,i)$ be a formalization of the concept of the i-th variable.

5.2 Proposition. There exists a Σ_0 formula $Value(y,x,z,u)$ and a polynomial p such that it is provable in $I\Sigma_0$ that:

(1) $$\begin{aligned} &Term(x_1)\,\&\,Term(x_2)\,\&\,|x_1|,|x_2| \leq v\,\&\,|u| > p(|\max(z)|,v) \rightarrow \\ &\quad\rightarrow (Var(x_1,i). \rightarrow Value(y,x_1,z,u) \equiv (z)_i = y) \\ &\quad\&\,(Value(y,\overline{0}^{\bullet},z,u) \rightarrow y = \overline{0}) \\ &\quad\&\,(Value(y_1,\mathbf{S}^{\bullet}(x_1),z,u)\,\&\,Value(y_2,x_1,z,u) \rightarrow y_1 = S(y_2)) \\ &\quad\&\,(Value(y,x_1{+}^{\bullet}x_2,z,u)\,\&\,Value(y_1,x_1,z,u) \\ &\qquad\&\,Value(y_2,x_2,z,u) \rightarrow y = y_1 + y_2) \\ &\quad\&\,(Value(y,x_1{*}^{\bullet}x_2,z,u)\,\&\,Value(y_1,x_1,z,u) \\ &\qquad\&\,Value(y_2,x_2,z,u) \rightarrow y = y_1 * y_2)\,; \end{aligned}$$

(2) $$\begin{aligned} &Term(x)\,\&\,|x| \leq v\,\&\,|u_1|,|u_2| \geq p(|\max(z)|,v) \\ &\&\,Value(y_1,x,z,u_1)\,\&\,Value(y_2,x,z,u_2) \rightarrow y_1 = y_2\,. \end{aligned}$$

Proof. $Value(y,x,z,u)$ is defined as $(\exists s \leq u)\varphi(y,x,z,s)$, where $\varphi(y,x,z,s)$ is a Σ_0 formula expressing the following conditions:

(1) s is a sequence of pairs $(a_0,b_0),\ldots,(a_m,b_m)$;
(2) for every $i \leq m$, either a_i is a variable, or a_i is $a_j{+}^{\bullet}a_k$, where $0 \leq j$, $k < i$, etc. for $\mathbf{S}, \overline{0}$ and $*$;
(3) if a_i is the j-th variable, then $b_i = (z)_j$; if a_i is $a_j{+}^{\bullet}a_k$, then $b_i = b_j + b_k$; etc. for $\mathbf{S}, \overline{0}$ and $*$;
(4) $y = b_m$ and $x = a_m$.

Clearly, if s satisfying these conditions exists, then it determines y uniquely and the conditions of the proposition are provable. (Namely, using induction, one can prove that every sequence of pairs for a subterm of t is a subsequence of some sequence of pairs for t.) Thus we only have to find an upper bound to the size of s. Assume s is given. By Proposition 1 (1),

$$b_i \leq (\max(z)+2)^{|a_i|}.$$

Since the pairing function is a polynomial, (a_i, b_i) is bounded by a polynomial in v and $(\max(z)+2)^v$, hence $(a_i, b_i) \leq (\max(z)+2)^{q(v)}$ for some polynomial q. By Proposition 3.30, $|s| \leq r(|(\max(z)+2)^{q(v)}|, v)$ for some polynomial r, which gives the required bound $|s| \leq p(|\max(z)|, v)$, p a polynomial.

The details can be worked out in a similar way as we did in Sect. 3. □

5.3 Corollary. The value of an L_0 term is definable by a Σ_0^{exp} formula (in fact by a bounded formula in L_2).

Proof. Define

$$val(x,z) = y \equiv_{df} Value(y, x, z, 2^{p(|\max(z)|,|x|)}). \qquad \square$$

The meaning of $\Gamma(x,z,u)$ in the following theorem is: the Σ_0 formula x is satisfied by a string of numbers z and u is some bound.

5.4 Theorem. There exists a Σ_0 formula $\Gamma(x,z,u)$ and a constant c such that it is provable in $I\Sigma_0$ that:

$$\begin{aligned}
|u| \geq (\max(z)+2)^{c^v} \,\&\, |x| \leq v. \rightarrow (x = t =^\bullet s \rightarrow (\Gamma(x,z,u) \equiv \\
(\exists y)(Value(y,t,z,u) \,\&\, Value(y,s,z,u)))) \,\&\, \\
\&\,(x = (x_1 \rightarrow^\bullet x_2) \rightarrow (\Gamma(x,z,u) \equiv (\Gamma(x_1,z,u) \rightarrow \Gamma(x_2,z,u)))) \,\&\, \\
\&\,(x = \neg^\bullet x_1 \rightarrow (\Gamma(x,z,u) \equiv \neg\Gamma(x_1,z,u))) \,\&\, \\
\&\,(x = (\exists^\bullet w \leq^\bullet t)x_1 \rightarrow (\Gamma(x,z,u) \equiv (\exists y, y')(Value(y',t,z,u) \,\&\, y \leq y' \\
\&\,\Gamma(x_1,z',u)))),
\end{aligned}$$

where we assume that w is the j-th variable for some j and sequence z' is obtained by replacing its j-th entry by y; the conditions (Tarski's conditions) for $\leq^\bullet$ and bounded $\forall^\bullet$ are similar and therefore omitted.

Proof. The proof is similar to the one above. We define $\Gamma(x,z,u)$ as $(\exists s \leq u)\psi(s)$, where $\psi(s)$ says that s is a sequence consisting of pairs of some substitution instances of subformulae and their truth values satisfying Tarski's conditions. Let s be the sequence of pairs $(a_0,b_0),\ldots,(a_m,b_m)$. We shall require the following condition. Let a_i be of the form $(\exists w \leq t)\varphi(w)$ or $(\forall w \leq t)\varphi(w)$. Then we include all formulae $\varphi(\overline{n})$ for $n \leq val(t,z)$ in $a_0,\ldots,a_m$. Here $\overline{n}$ denotes the *dyadic* numeral introduced in Sect. 3(g). (If the value of $(\exists^\bullet w \leq^\bullet t)\varphi(w)$ turns out to be TRUE, then it suffices to include just one $\varphi(\overline{n})$, but this is not a real advantage.) By Proposition 1(2), the values needed to compute the truth value of a formula x are bounded by

$$(\max(z)+2)^{2^{|x|}}.$$

Thus the a_i's are Gödel numbers of subformulae of x with some variables replaced by numerals $\overline{n}$ for

$$n \leq (\max(z)+2)^{2^{|x|}}.$$

The length of such formulae is estimated by

$$c_1 * |(\max(z)+2)^{2^{|x|}}| * |x|,$$

where c_1 is a constant, since the length of a numeral is linear in its value. The length of (a_i, b_i) is thus

$$c_2 * |(\max(z)+2)^{2^{|x|}}| * |x|,$$

for some constant c_2. Now we estimate the number m of such pairs. We have at most $|x|$ variables in x, and each variable can be replaced by a numeral $\overline{n}$ with the bound above for n. This gives $((\max(z)+2)^{2^{|x|}}+1)^{|x|}$ possibilities and there are at most $|x|$ subformulae, hence

$$m \leq |x| * ((\max(z)+2)^{2^{|x|}}+1)^{|x|}.$$

Thus we can estimate the length of s by

$$c_3 * |(\max(z)+2)^{2^{|x|}}| * |x|^2 * ((\max(z)+2)^{2^{|x|}}+1)^{|x|} \leq (\max(z)+2)^{c^{|x|}},$$

where c_3 and c are suitable constants. Hence if u is larger than or equal to this bound, then there exists some sequence s that witnesses the truth of x. The provability of Tarski's conditions is shown as above. □

For a string of variables $z_1, \ldots, z_n$ we denote by $(z_1, \ldots, z_n)$ the code of the sequence $z_1, \ldots, z_n$.

5.5 Corollary. (1) For every bounded formula $\alpha(z_1, \ldots, z_n)$ in L_0 there exists a constant k such that

$$I\Sigma_0 \vdash u \geq 2^{(\max(z_1,\ldots,z_n)+2)^k} \rightarrow (\alpha(z_1, \ldots, z_n) \equiv \Gamma(\overline{\alpha}, (z_1, \ldots, z_n), u)).$$

(2) Let M be a model of $I\Sigma_0$, let $a, b, d \in M$, let $\alpha(x)$ be a Σ_0 formula. Suppose d is nonstandard and $b > 2^{(a+2)^d}$. Then

$$M \vDash \alpha(a) \equiv \Gamma(\overline{\alpha}, (a), b).$$

Proof. (1) First apply Proposition 5.2 to prove the statement for α atomic using induction on the complexity of terms in it. Then for general α use induction on the logical complexity of α and Theorem 5.4.

(2) This is an immediate consequence of (1). □

5.6 Theorem. $I\Sigma_0 + Exp$ is finitely axiomatizable.

Proof. Let T be a finite fragment $I\Sigma_0$ such that T proves Tarski's conditions of Theorem 5.4 for Γ. Then for every α bounded, T proves the formula of Corollary 5.5 (1). Let φ be defined by

$$\varphi(x, y, p, u) \equiv \Gamma(x, (y, p), u) .$$

(Here we are misusing notation: to be quite precise, we should distinguish between a *pair* and a *two element sequence*, but we do not want to introduce new notation.) Let T' be T plus the least number principle for $\varphi(x, y, p, u)$, where y is the induction variable and x, p and u are parameters, plus Exp. Let $\alpha(y, p)$ be an arbitrary bounded formula for which we want to prove the least number principle (clearly it suffices to have just one parameter). We shall argue in T'. Let p be given and suppose that for some y_0 we have $\alpha(y_0, p)$. Since we have Exp, we can take

$$u \geq 2^{(\max(y_0, p))^k} ,$$

where k corresponds to α. Hence, for $y \leq y_0$,

$$\alpha(y, p) \equiv \Gamma(\overline{\alpha}, (y, p), u) \equiv \varphi(\overline{\alpha}, y, p, u) .$$

Thus we can take the least y for $\varphi(\overline{\alpha}, y, p, u)$. □

(c) An Interpretation of $I\Sigma_0$ in Q

Q is a very weak theory, e.g. the associativity of $+$ is not provable in it. However from the point of view of interpretability it is quite strong. We shall show that $I\Sigma_0 + \Omega_n$ is interpretable in Q for every n. Moreover the form of interpretation is very natural: we take the same operations and only restrict the domain. The domain of interpretation is, roughly speaking, an initial segment of the original numbers. This suggests an attractive finitist's program, which was pursued by Edward Nelson [Nelson 86]. Namely, the consistency of Q seems quite evident; further it is also evident that interpretation preserves consistency, thus if we develop mathematics only in theories interpretable in Q, we are safe-guarded against inconsistencies. Later on we shall see that we can even interpret in Q some consequences of $I\Sigma_0 + Exp$ which are not available in $I\Sigma_0 + \Omega_n$ (Theorem 5.27 (i)).[1]

[1] Let us remark, for those who are interested in minimal foundations of mathematics, that one can use an extremely weak system for set theory instead of Q, since Q is interpretable in it. The system has only two simple axioms:

$$(\exists x)(\forall y)(\neg y \in x) ;$$

$$(\forall x)(\forall y)(\exists z)(\forall u)(u \in z \equiv (u \in x \vee u = y)) .$$

The main reason why we consider interpretation of Bounded Arithmetic in Q is, however, different. The interpretability of systems of Bounded Arithmetic gives us *equiconsistency* of them with Q, and this equiconsistency is provable in Bounded Arithmetic too. Hence Gödel theorem implies that such a system does not already prove consistency of Q.

The rest of the subsection is almost entirely devoted to the proof of the following theorem. In the whole subsection we shall use only interpretations given by restricting the domain. Such an interpretation is determined by a formula with one free variable.

5.7 Theorem. For every n, there exists a global interpretation of $I\Sigma_0 + \Omega_n$ in Q.

By a *global* interpretation we mean the usual concept of interpretation. The word *global* is used here to stress the fact that we have one translation of the language (but in fact only the domain will be different) such that the translations of each one of the infinitely many axioms are provable. On the other hand, *local* interpretation means that we can choose a different translation for every finite subset of axioms. Nelson uses local interpretations in his book. We shall use a local interpretation as an intermediate step.

We shall say that a formula $I(x)$ is *inductive* in a theory T if

$$T \vdash I(\overline{0}) \,\&\, (\forall x)(I(x) \to I(S(x)))\,.$$

Recall that I is a *cut* in T if it satisfies moreover

$$I(x) \,\&\, y \leq x \to I(y)\,,$$

provably in T. We shall say that $J(x)$ is a *subcut* of $I(x)$ in T if moreover

$$T \vdash J(x) \to I(x)\,.$$

Let Q^+ be Q augmented with the following axioms:

(associativity of +) $\quad (x+y)+z = x+(y+z)\,;$

(left distributivity) $\quad x(y+z) = xy + xz\,;$

*(associativity of *)* $\quad (xy)z = x(yz)\,.$

5.8 Lemma. There exists an inductive formula $I(x)$ which determines an interpretation of Q^+ in Q.

We shall not prove this lemma. It can be proved using similar tricks as we shall use below, but it is technically more complicated, since we start with a very weak theory Q. A complete proof can be found in [Nelson 86].

5.9 Lemma. Let $I(x)$ be an inductive formula in Q. Then there exists a subcut $J(x)$ of $I(x)$ in Q.

Proof. We shall use the following three easy theorems of Q:
(i) $x \leq \overline{0}$,
(ii) $x \leq y \equiv S(x) \leq S(y)$,
(iii) $\overline{0} \leq x$.
Let I be inductive in Q. Let K be defined by

$$K(x) \equiv I(x) \,\&\, (\forall y, z)(z \leq y \,\&\, y \leq x \rightarrow z \leq x)\,.$$

We shall show that K is inductive. $K(\overline{0})$ follows easily from (i) and from $I(\overline{0})$. Suppose $K(x)$. Then, clearly, $I(S(x))$ and it remains to prove the second part of the condition for $K(S(x))$. Let $z \leq y \leq S(x)$. If $z = \overline{0}$, then, by (iii), $z \leq x$. Otherwise, by (Q3), $z = S(z')$, for some z'. Then, by (i), (Q5) and (Q1), y cannot be $\overline{0}$. Thus $y = S(y')$, for some y'. Thus we get $z' \leq y' \leq x$, using (ii). By $K(x)$, we have $z' \leq x$. Using (ii) once again, we get $z = S(z') \leq S(x)$. Hence K is inductive.

Now define

$$J(x) \equiv (\forall y \leq x)K(y)\,.$$

First we shall show that J is inductive. $J(\overline{0})$ follows easily from (i). Assume $J(x)$ and let $y \leq S(x)$. We need to show that $K(y)$. If $y = \overline{0}$, then $K(y)$, since K is inductive. Suppose that $y = S(y')$, for some y'. Then, by (ii), we get $y' \leq x$, hence $K(y')$. Since K is inductive, $K(S(y'))$, which is $K(y)$. Thus K is inductive. Now we shall prove that

$$J(x) \,\&\, y \leq x \rightarrow J(y)\,.$$

Suppose $J(x) \,\&\, y \leq x$. Let $z \leq y$ be arbitrary. We need to prove $K(z)$. By $J(x)$, we have in particular $K(x)$, hence $z \leq x$. Thus we have $K(z)$, since $J(x)$. □

5.10 Lemma. Let $I(x)$ be an inductive formula in Q^+. Then there exists a cut $J(x)$ such that J is a subcut of I closed under S, $+$ and $*$, i.e. Q^+ proves

$$J(x) \rightarrow I(x)\,;$$
$$J(\overline{0})\,;$$
$$J(x) \,\&\, J(y) \rightarrow J(S(x)) \,\&\, J(x+y) \,\&\, J(xy)\,.$$

Proof. By Lemma 5.9 we may assume that I is a cut. We shall use the same construction as in the proof of Theorem 3.5, Chap. III. Take

$$J_0(x) \equiv (\forall y)(I(y) \rightarrow I(y+x))\,;$$
$$J(x) \equiv (\forall y)(J_0(y) \rightarrow J_0(yx))\,.$$

Then J_0 contains $\overline{1}$, since I is inductive; J_0 is closed under $+$ by associativity of $+$; J is contained in J_0 (since J_0 contains $\overline{1}$) and hence also in I; J is closed under $+$ by left distributivity and since J_0 is closed under $+$; J is closed under S, since it contains $\overline{1}$ and it is closed under $+$; J is closed under $*$ by associativity of $*$. To show that J is closed downwards, we use the assumption that I is a cut, left distributivity and J_0 being closed under $+$. □

5.11 Lemma. $I\Sigma_0$ is locally interpretable in Q.

Proof. By Lemma 5.8 we can take Q^+ instead of Q. Let $\varphi_1(x,\mathbf{p}),\dots,\varphi_n(x,\mathbf{p})$ be given bounded formulae. We want to interpret Q plus induction for $\varphi_1(x,\mathbf{p}),\dots,\varphi_n(x,\mathbf{p})$ in Q^+. Let

$$I_j(x,\mathbf{p}) \equiv \varphi_j(\overline{0}) \,\&\, (\forall y \le x)(\varphi_j(y,\mathbf{p}) \to \varphi_j(S(y),\mathbf{p})) \to \varphi_j(x,\mathbf{p}),$$

for $j = 1,\dots,n$. Let

$$I(x) \equiv (\forall \mathbf{p})(I_1(x,\mathbf{p}) \,\&\, \dots \,\&\, I_n(x,\mathbf{p})).$$

Since each $I_j(x,\mathbf{p})$ is inductive with respect to x in Q^+, so is $I(x)$. By Lemma 5.10 we can find a subcut $J(x)$ closed under $+$ and $*$. Since J is closed under S, $+$ and $*$ and it is closed downwards, we have, for any bounded formula $\alpha(z_1,\dots,z_k)$,

$$Q^+ \vdash J(z_1) \,\&\, \dots \,\&\, J(z_k) \to (\alpha(z_1,\dots,z_n) \equiv (\alpha(z_1,\dots,z_k))^J).$$

Since

$$J(x) \to I(x) \to I_j(x)$$

provably in Q^+, for every j, J determines an interpretation of induction for every φ_j. All but two axioms of Q (even Q^+) are open, hence they hold in the interpretation. The remaining two axioms are (Q3): $x \neq \overline{0} \to (\exists y)(x = S(y))$ and (Q8): the definition of the relation $\le$. The first one is provable using induction for bounded formulae from the others (hint: first prove $x \le S(x)$, then $x \neq \overline{0} \to (\exists y \le x)(x = S(y))$). Hence we can simply suppose that we have enough formulae amongst $\varphi_1,\dots,\varphi_n$ to prove it. The last one causes no problems, since we can trivially interpret Q in the theory obtained from Q by eliminating $\le$ from the language and deleting (Q8). □

Proof of Theorem 5.7. By Theorem 3.5, Chap. III, we know that each cut in a sufficiently strong theory can be shortened to a cut closed under ω_n, for a given n. One can easily check that $I\Sigma_0$ is strong enough for that. So, by the last lemma, we only have to interpret $I\Sigma_0$ in a finite fragment T of $I\Sigma_0$. We take T so strong that

(1) the properties of the exponentiation relation are provable in it,
(2) Tarski's conditions for the formula Γ of Theorem 5.4 are provable in it, and
(3) the least number principle for $\Gamma(x,(y,p),u)$, with y as the induction parameter, is provable in it.

By the same theorem of Chap. III on shortening cuts we get a cut $I(x)$ such that

$$T \vdash I(x) \to (\exists y)(y = 2^{2^x}).$$

By Lemma 5.10, we can assume that I is closed under $+$ and $*$. We claim that I determines an interpretation of $I\Sigma_0$. It suffices to prove the least number principle for every bounded $\alpha(y,p)$ with just one parameter. Let p be an element in I and suppose that for some y_0 in I we have $\alpha(y_0,p)$. Take

$$u = 2^{(\max(y_0,p))^k},$$

where k corresponds to α. This is possible, since

$$2^{(\max(y_0,p))^k} \leq 2^{2^{\max(y_0,p)*k}},$$

and the last number exists by a property of I. Now we conclude as in Theorem 5.6: for $y \leq y_0$,

$$\alpha(y,p) \equiv \Gamma(\overline{\alpha},(y,p),u),$$

and we can take the least y for $\Gamma(\overline{\alpha},(y,p),u)$. □

The concept of consistency *Con* used in the following theorem is the usual one as used in Chap. III, except that we must use the efficient coding of sequences as described in Sect. 3 of this section.

5.12 Theorem. For every n,

$$I\Sigma_0+\Omega_1 \vdash Con(I\Sigma_0+\Omega_n^{\bullet}) \equiv Con(Q^{\bullet}).$$

Proof. The consistency of $I\Sigma_0 + \Omega_n$ implies the consistency of Q trivially. The consistency of $I\Sigma_0 + \Omega_n$ can be reduced to the consistency of Q using an interpretation. We shall analyze this proof, in order to see that it can be done in $I\Sigma_0 + \Omega_1$. Let a proof of a contradiction in $I\Sigma_0 + \Omega_n$ be given. Using the interpretation constructed above we can transform it into a proof of a contradiction in Q. Again, it is clear that the interpretation as a relation can be described by a Σ_0 formula. Hence we only have to check that the size of the transformed proof is polynomial in the original one. This can be proved by showing that each application of a logical rule in the original proof is replaced by only a polynomially longer sequence in the new proof. There

is only one place where we have to modify the interpretation given above. When we prove the least number principle for a bounded formula α, we define parameter u by

$$u = 2^{(\max(y_0,p))^k}.$$

Now we have to take into account how k depends on α, since α is not fixed. Thus instead of the double exponential function in the definition of $I(x)$, we take triple exponential. Hence

$$Q \vdash I(x) \to (\exists y)\left(J(y) \,\&\, y = 2^{2^{2^x}}\right),$$

where cut $J(y)$ determines an interpretation of a sufficiently strong finite fragment T of $I\Sigma_0$ in Q. Then we replace number k in the definition of u by a more explicit bound given by Theorem 5.4:

$$u = 2^{(\max(y_0,p))^{c|\alpha|}},$$

where c is the constant independent of α from Theorem 5.4. Now, using the estimate

$$2^{(\max(y_0,p))^{c|\alpha|}} \leq 2^{2^{2^{\max(y_0,p)+c|\alpha|}}}$$

one can show that u exists and $J(u)$ holds true, and the proofs of these facts in Q are of polynomial length in the length of α. Hence the induction axioms are replaced also by polynomially long sequences. □

(d) Cut-Elimination and Herbrand's Theorem in Bounded Arithmetic

There are two basic measures of the complexity of proofs: the length of proofs and the quantifier complexity of proofs. Cut-elimination and Herbrand's theorem implies that each first order tautology α has a proof whose quantifier complexity is not larger than the quantifier complexity of α. The cost for the reduction of the quantifier complexity is an increase in the size of the proof. This is the reason why this theorem is not provable even in $I\Sigma_0 + Exp$ in full generality. Nevertheless there is a restricted version of this result which is provable in $I\Sigma_0 + Exp$ and which has several applications. We shall prove it in this subsection. The proof will be just an analysis of a classical proof of cut-elimination. There is an alternative proof based on Hilbert style calculi, but we shall use a variant of Gentzen's approach since it is simpler and more transparent. The transformations used in the proof will again be some simple manipulations with sequences, hence definable by Σ_0 formulae. Thus we only have to estimate the size of constructed proofs.

We shall use a slightly simplified version of Schwichtenberg's system [Schwichtenberg 77] and follow his proof of cut-elimination. The proofs will

be only briefly sketched; however we do not presuppose knowledge of proof theory. For more details the reader should consult [Takeuti 80]. It should be stressed here that the translations from this system to the system introduced in Chap. 0 and backwards are quite elementary. In particular (assuming efficient coding, which was introduced in Sect. 3), it is possible to prove in $I\Sigma_0 + \Omega_1$ that a sentence is provable in one system iff its translation is provable in the other one.

*

We consider a system for first order logic. Its language consists of logical connectives $\&$, $\vee$, $\neg$, quantifiers $\forall$, $\exists$, variables, constants, function symbols and relation symbols. We do not distinguish equality as a special (or logical) relation symbol. In order to reduce the number of rules, and consequently the number of cases to be considered in the proofs, we allow negation only at atomic formulae. Thus, for a complex formula φ, the expression $\neg\varphi$ is an abbreviation for the equivalent formula obtained by applying De Morgan rules. For a similar reason we shall treat disjunction $\vee$ differently too. We shall think of the formulae in a disjunction as forming a set rather than a sequence. In the arithmetization, all syntactical objects are represented as sequences (which in turn are represented as binary expansions). Thus what we mean is that: *in proofs we allow the replacement of a disjunction by another one which has the same elements.* For reasons of symmetry we shall consider conjunction $\&$ to be also a set operation. When we write, for instance, that a formula φ has the form $\gamma \vee \psi$, we mean that φ is the disjunction of a set $\{\gamma_1, \dots, \gamma_m, \psi_1, \dots, \psi_n\}$, where $\gamma_1, \dots, \gamma_m, \psi_1, \dots, \psi_n$ are atomic formulae, conjunctions or formulae starting with a quantifier, and we do not assume that the sets $\{\gamma_1, \dots, \gamma_m\}$, $\{\psi_1, \dots, \psi_n\}$ are disjoint and γ is the disjunction of $\gamma_1, \dots, \gamma_m$, or $\gamma = \gamma_1$ if $m = 1$, and ψ is the disjunction of $\psi_1, \dots, \psi_n$, or $\psi = \psi_1$ if $n = 1$.

A *proof* is a sequence of formulae (usually called *proof lines*, or *steps*) where each formula is either a logical axiom or follows from preceding ones by a rule.

There is one axiom schema in our system:

(A) $\quad \gamma \vee \varphi \vee \neg\varphi,$

and five rules:

(W) $\quad \dfrac{\gamma}{\gamma \vee \varphi}, \qquad (\&)\ \dfrac{\gamma \vee \varphi,\ \gamma \vee \psi}{\gamma \vee (\varphi \,\&\, \psi)},$

($\forall$) $\quad \dfrac{\gamma \vee \varphi(x)}{\gamma \vee (\forall y)\varphi(y)}$ where the variable x is not free in γ,

($\exists$) $\quad \dfrac{\gamma \vee \varphi(\tau)}{\gamma \vee (\exists x)\varphi(x)}$ where τ is a term free for x in φ,

(Cut) $\dfrac{\gamma \vee \varphi,\ \delta \vee \neg\varphi}{\gamma \vee \delta}$.

The elements of the disjunctions γ and δ are called *side formulae*; γ and δ may consist of one formula or can even be void. In *Cut*, φ and $\neg\varphi$ are *cut formulae*. Let us point out that an important rule, *the contraction rule*, is only implicit in our system. A typical application of it, which is also needed in the proof of Herbrand's theorem, is the following: we derive by $(\exists)$

$$\frac{\gamma \vee (\exists x)\varphi(x) \vee \varphi(\tau)}{\gamma \vee (\exists x)\varphi(x)}.$$

In the lower sequent we do not repeat $(\exists x)\varphi(x)$ once more, since our disjunctions are *sets*. Rule (W) is clearly redundant (and in contrast with (Cut) also harmless), but it is convenient to have it in the system.

The fact that we define proofs to be *sequences*, not *trees* (the latter being common in proof theory), will be important later, when we shall consider systems weaker than $I\Sigma_0 + Exp$.

We shall follow the notation of Chap. IV, Sect. 4 by writing $Proof(d, \gamma)$ for the relation *d is a proof of* γ. By writing $\gamma(x)$ and then $\gamma(\tau)$, we denote the substitution of τ for all free occurrences of x in γ. We shall use two measures of the complexity of proofs. The *length*, denoted $lh(d)$, is the length of the sequence which codes the proof d. Of course, lh is defined also for formulae and terms. We assume that the number of symbols used in the coding sequences is finite, thus the Gödel number of a proof d is bounded by $c^{lh(d)}$, for some constant c. Further we define, for a formula φ, $rank(\varphi)$ as the number of connectives and quantifiers in φ plus 1. The *cut-rank* of a proof d, denoted by $c\text{-}rank(d)$, is 0 if there is no cut in it; otherwise it is the maximum of the ranks of the cut formulae.

5.13 Lemma $(I\Sigma_0 + Exp)$. (i) Suppose $Proof(d, \gamma \vee (\forall y)\psi(y))$; then there exists d' such that $Proof(d', \gamma \vee \psi(x))$ and $c\text{-}rank(d') \leq c\text{-}rank(d)$, $lh(d') \leq lh(d)$.

(ii) If $Proof(d(x), \gamma(x))$, then for any term τ, $Proof(d(\tau), \gamma(\tau))$ and $c\text{-}rank(d(x)) = c\text{-}rank(d(\tau))$, $lh(d(\tau)) \leq lh(d(x)) * lh(\tau)$.

(iii) Suppose $Proof(d, \varphi_0 \,\&\, \varphi_1)$, then there exist d_0, d_1 such that $Proof(d_i, \varphi_i)$ and $c\text{-}rank(d_i) \leq c\text{-}rank(d)$, $lh(d_i) \leq lh(d)$, for $i = 0, 1$.

(We take x and τ in (i) and (ii) so that there are no clashes of variables.)

Intuitively, the statements are clear; formally they can be proved by induction on the length of the proof. The proofs are easy, therefore we omit them.

5.14 Reduction Lemma $(I\Sigma_0 + Exp)$. Suppose d is a proof of Θ such that the last inference is a cut with a cut formula φ. Suppose that the part of the proof before the last inference has cut-rank less than $rank(\varphi)$. Then there exists d', a proof of Θ, such that $c\text{-}rank(d') < c\text{-}rank(d)$ and $lh(d') \leq lh(d)^3$.

Proof. Let the last inference be

$$\frac{\gamma \vee \varphi,\ \delta \vee \neg\varphi}{\gamma \vee \delta}\,.$$

1. First suppose that φ is atomic. Then $rank(\varphi) = 1$, hence there is no other cut in the proof. The proof d contains a proof d_0 of $\gamma \vee \varphi$ of rank 0. Consider how φ enters the proof d_0. If it is by (W) or as a side formula in (A), then we can omit it and derive $\gamma \vee \delta$ by (W). If it is by (A) of the form $\sigma \vee \varphi \vee \neg\varphi$, then we first derive $\sigma \vee \delta \vee \neg\varphi$ from $\delta \vee \neg\varphi$ by weakening (W) and then use this proof instead of the proof line $\sigma \vee \varphi \vee \neg\varphi$. We remove φ also from the proof lines below $\sigma \vee \varphi \vee \neg\varphi$ and extend them by forming disjunctions with δ. Thus we transform the proof of $\gamma \vee \varphi$ into a proof d' of $\gamma \vee \delta$.

2. Suppose φ is $(\exists x)\psi(x)$. Our strategy is the same as above, but now φ can be introduced also by rule $(\exists)$. (Note that φ can be introduced in d several times by $(\exists)$; see the example above describing the contraction.) Consider such a derivation in d:

$$\frac{\sigma \vee \psi(\tau)}{\sigma \vee (\exists x)\psi(x)}\,,$$

where it is possible that the lower disjunction is just σ. In d we have a proof of $\delta \vee \neg\varphi$, which is $\delta \vee (\forall x)(\neg\psi(x))$. By Lemma 5.13 (i) and (ii) we get a proof of $\delta \vee \neg\psi(\tau)$. Now we apply cut to $\sigma \vee \psi(\tau)$ and $\delta \vee \neg\psi(\tau)$ to get $\sigma \vee \delta$. This cut has smaller rank, hence, after all these changes, we get a proof d' of $\gamma \vee \delta$ of a smaller rank.

3. If φ is a conjunction $\varphi_0 \,\&\, \varphi_1$, then we do the same procedure as in 2 except that we now use Lemma 5.13 (iii).

4. By duality there are no other cases. If, for instance, φ is a disjunction, then $\neg\varphi$ is a conjunction.

5. It remains to compute the size of the transformed proof d'. This will also be a proof that the above considerations can be carried out in $I\Sigma_0 + Exp$. The proof d' is constructed in such a way that for certain proof lines of d we add a proof of $\delta \vee \neg\varphi$ or a proof of $\delta \vee \neg\psi(\tau)$, or a proof of $\delta \vee \varphi_i$ or just extend the disjunction by δ and remove φ from it. The largest increase in the size occurs when we need the proof of $\delta \vee \neg\psi(\tau)$; there we have the following upper bound from Lemma 5.13:

$$lh(d) * lh(\tau) \leq lh(d)^2\,.$$

As there are at most $lh(d)$ places where we do such extensions, the bound is $lh(d)^3$. □

5.15 Lemma $(I\Sigma_0 + Exp)$. If $Proof(d, \gamma)$ and $c\text{-}rank(d) > 0$, then there exists d' such that $Proof(d', \gamma)$, $c\text{-}rank(d') < c\text{-}rank(d)$ and $lh(d') \leq 2^{2^{2 * lh(d)}}$.

Proof. Take the initial segment of proof d down to the first cut whose rank is equal to $c\text{-}rank(d)$. Replace this subproof by a proof whose rank is less than $c\text{-}rank(d)$ using Lemma 5.14. Repeat this procedure until no cut with rank equal to $c\text{-}rank(d)$ remains. We start with a proof of length $lh(d)$; then in each step the length is raised to the 3, and there are at most $lh(d)$ cuts that we have to eliminate. This gives

$$lh(d') \leq lh(d)^{3^{lh(d)}} \leq 2^{2^{2*lh(d)}}. \qquad \square$$

Now we shall define several concepts of provability and consistency. The basic concepts have already been defined in Chap. III; we repeat their definition for the reader's convenience.

5.16 Definition.

(0) *d is a proof of x in T*

$$Proof_T(d,x) \equiv_{df} (\exists s \leq d)((\forall i < lh(s))T((s)_i)\,\&\, \\ \&\, Proof(d,(s)_0\,\&^\bullet \ldots \&^\bullet (s)_{lh(s)-1} \rightarrow^\bullet x)).$$

(1) *Provability in the theory T*

$$Pr_T(x) \equiv_{df} (\exists d) Proof_T(d,x).$$

(2) *Restricted provability in the theory T*

$$RPr_T(k,x) \equiv_{df} (\exists d)(c\text{-}rank(d) \leq k\,\&\, Proof_T(d,x)).$$

(3) *Cut-free provability in the theory T*

$$CFPr_T(x) \equiv_{df} RPr_T(\overline{0},x).$$

(4) *Herbrand provability in the theory T*

$$HPr_T(x) \equiv_{df} \text{“there exists a propositional proof of a disjunction of instances of the open part of } He(\bigwedge S \rightarrow x), \text{ for some finite } S \subseteq T''\text{,}$$

(see 0.18 for the definition of the function He).

(5) *Consistency of the theory T*

$$Con_T \equiv_{df} \neg Pr_T(\overline{\overline{0} = \overline{1}}).$$

(6) *Restricted consistency of the theory T*

$$RCon_T(k) \equiv_{df} \neg RPr_T(k, \overline{\overline{0} = \overline{1}}).$$

(7) *Cut-free consistency of the theory T*

$$CFCon_T \equiv_{df} \neg CFPr_T(\overline{\overline{0} = \overline{1}}).$$

(8) *Herbrand consistency of the theory T*

$$HCon_T \equiv_{df} \neg HPr_T(\overline{\overline{0} = \overline{1}}).$$

(If we omit the subscript, it means *provability* or *consistency* in pure logic.)

The theories that we shall use will be either finite or have naturally defined polynomial time computable sets of axioms. For such theories T the predicate "φ is an axiom of T" can be expressed by a Σ_1^b formula. Then $Proof_T(d, x)$ is also a Σ_1^b formula.

Denote by *Superexp* an axiom saying that the Σ_0 definable function 2_y^x is total (see Definition III.3.3).

5.17 Theorem

(i) $I\Sigma_0 + Superexp \vdash (\forall x)(Pr^\bullet(x) \to CFPr^\bullet(x))$.
(ii) For every k, $I\Sigma_0 + Exp \vdash (\forall x)(RPr^\bullet(\overline{k}, x) \to CFPr^\bullet(x))$.

Proof. Both statements follow immediately from Lemma 5.15. □

Let us note that it is provable in $I\Sigma_0 + Exp$ that each *propositional* tautology has a cut-free proof. This can be shown by formalizing a completeness proof for the propositional fragment of our system consisting of the schema (A) and rule (&). Let a propositional tautology φ be given. Construct a proof of φ by applying rule (&) backwards, i.e. we split conjunctions into parts until only disjunctions of atomic and negated atomic formulae are left (recall that negations can occur only at atomic formulae). We must show that these disjunctions are of the form (A) which just means that they are tautologies. This is true because rule (&) is sound also in the opposite direction. In each step of this construction we add two formulae of length at most $lh(\varphi)$ and the construction has at most $lh(\varphi)$ steps. Thus the total size of the proof is estimated from above by $lh(\varphi) * 2^{lh(\varphi)}$. Therefore, the construction can be carried out in $I\Sigma_0 + Exp$.

* *

We conclude this subsection by showing that in $I\Sigma_0 + Exp$ the cut-free provability and the Herbrand provability are equivalent. Then we derive corollaries about the provability of the consistency of Q.

5.18 Middle Sequent Theorem. Let φ be a first order tautology in *prenex normal form*. Then there exists a proof d of φ which is cut-free and such that it can be divided into two parts d' and d'' such that d' is quantifier free, while d'' contains only applications of the quantifier rules.

The idea of the **proof** is simple. First one shows that axiom schema (A) and rule (W) can be restricted so that the newly introduced formulae must be quantifier-free. This increases the length of proofs only polynomially and no new cuts are needed. Now take such a cut-free proof of φ and switch quantifier and non-quantifier rules to obtain the required form of a proof. The only thing that we have to take care of are possible collisions of variables in rule ($\forall$). Therefore we must first transform the proof into a *regular proof*, which is a proof in which each application of rule ($\forall$) has its own variable that is generalized (*eigenvariable*). To do such a renaming we have to transform the proof into tree form, which means that each premise is used only once. This may cause at most an exponential increase. Hence it is provable in $I\Sigma_0 + Exp$ that a formula φ has such a proof whenever it has a cut-free proof.

Suppose such a proof d of φ is given. Let d' be the quantifier-free initial segment of the proof, let φ' be its last formula. We may also suppose that each formula, except the last one, is a premise of some application of a rule in d. Hence φ is obtained from φ' by successive applications of quantifier rules. In this situation we call φ' the *middle formula* of d. (Usually in proof theory sequents are used instead of our use of disjunction of formulae; therefore the theorem is referred to as the Middle Sequent Theorem.) We shall show that φ' can be easily transformed into a tautological Herbrand disjunction $He(\varphi)$.

Let φ be of the form

$$(\exists x_1)(\forall y_1)\dots(\exists x_n)(\forall y_n)\psi(x_1, y_1, \dots, x_n, y_n)\,.$$

We can transform any formula in a prenex normal form into such a form by adding dummy quantifiers. Then φ' has the following form

$$\bigvee_i \psi(t_1^i, z_1^i, \dots, t_n^i, z_n^i)\,,$$

where t_j^i are some terms and z_j^i are variables. Let $f_1, \dots, f_n$ be new function symbols that will correspond to the universally quantified variables of φ. Now we follow the derivation of φ from φ', but instead of applying the rules, we substitute terms in φ' as follows. If generalization ($\forall$) is applied to the variable z_j^i, then we substitute $f_j(t_1^i, \dots, t_j^i)$ for z_j^i. If the rule ($\exists$) is used, then we

do nothing. In order to prove that the resulting disjunction is a Herbrand disjunction, we shall show two things:

(i) $t_1^i, \ldots, t_j^i$ are changed (i.e. something is substituted in them) only after the generalization of z_j^i;

(ii) if $z_j^i = z_{j'}^k$, then $j = j'$ and $t_1^i = t_1^k, \ldots, t_j^i = t_j^k$.

Suppose (i) is false. Then some z_m^k which is generalized before z_j^i must occur in $t_1^i, \ldots, t_j^i$. Clearly z_m^k is not among $z_1^i, \ldots, z_j^i$. Hence before z_m^k can be generalized, its occurrences in $t_1^i, \ldots, t_j^i$ must be killed by existentially quantifying the terms in which it occurs. But this is a contradiction, since such existential quantifications are done only after the universal quantification of z_j^i.

Let us prove (ii). If z_j^i occurs in different disjuncts, they must be first contracted into one before we can generalize (see the restriction on $(\forall)$). Two different disjuncts may become identical in the course of the derivation d'' after we replace some terms and free variables in them by bounded variables. However, different $t_1^i, \ldots, t_j^i$ and $t_1^k, \ldots, t_j^k$ can be identified in such a way only after we quantify z_j^i (by the same argument as in (i)). Thus (ii) must be true.

Now it is clear that the procedure changes φ' gradually into a Herbrand disjunction. Condition (i) ensures that the terms have the required form, condition (ii) ensures that we replace the same variables by the same terms, hence we preserve the property of being a tautology. In each step the size of the disjunction increases quadratically (since we substitute some substrings of a string into itself), hence the size of the resulting $He(\varphi)$ can be estimated by

$$lh(d)^{2^{lh(d)}} \leq 2^{2^{2 \cdot lh(d)}}.$$

The converse transformation, from $He(\varphi)$ into a cut-free proof of φ, is also easy. Consider a formula φ of the same form as above. First eliminate the new function symbols $f_1, \ldots, f_n$ in $He(\varphi)$ by replacing maximal terms starting with such a function symbol by new variables. Let us denote the resulting formula by φ'. Since $He(\varphi)$ is a *propositional* tautology, so is φ'. Take some proof of φ'; we have shown above that there is always one that is cut-free and at most exponentially long. Now we only have to see that we can apply the quantifier rules in a suitable order to suitable terms and variables to get φ. The following rules lead to a proof of φ:

(1) Whenever we have a disjunct of the form

$$(\forall y_j)(\exists x_{j+1})(\forall y_{j+1}) \ldots (\exists x_n)(\forall y_n) \psi(t_1, z_1, \ldots, t_j, y_j, x_{j+1}, y_{j+1}, \ldots, x_n, y_n),$$

apply $(\exists)$ to the term t_j;

(2) if the above is not possible, then consider formulae of the form

$$(\exists x_{j+1})(\forall y_{j+1})\dots(\exists x_n)(\forall y_n)\\ \psi(t_1, z_1, \dots, t_j, z_j, x_{j+1}, y_{j+1}, \dots, x_n, y_n);$$

take one of them such that the term in $He(\varphi)$ to which z_j corresponds is of maximal length, and generalize z_j.

If we repeat this rule the disjunctions clearly converge to φ. Thus we only have to show that the rule can be applied repeatedly, until the disjunction becomes φ. For $(\exists)$ there are no restrictions, for $(\forall)$ we have to show that z_j occurs in only one disjunct and it does not occur in $t_1, z_1, \dots, t_{j-1}, z_{j-1}, t_j$ of this disjunct. The last thing is clear, since $z_1, \dots, z_j$ correspond to the terms $f_1(t_1), \dots, f_j(t_1, \dots, t_j)$, hence $f_j(t_1, \dots, t_j)$ cannot be a subterm of any of the terms

$$t_1, f_1(t_1), \dots, t_{j-1}, f_{j-1}(t_1, \dots, t_{j-1}), t_j\,.$$

Now suppose (1) is not possible and suppose z_j occurs in another disjunct. Since (1) is not possible, the disjunct must have the following form

$$(\exists x_{k+1})(\forall y_{k+1})\dots(\exists x_n)(\forall y_n)\\ \psi(t'_1, z'_1, \dots, t'_k, z'_k, x_{k+1}, y_{k+1}, \dots, x_n, y_n)\,.$$

The term corresponding to z'_k is $f_k(t'_1, \dots, t'_k)$. If z_j occurred in $t'_1, z'_1, \dots, t'_k$, then the term corresponding to z'_k would be longer than the one corresponding to z_j. Hence we can have only $z_j = z'_k$. But then $f_j(t_1, \dots, t_j) = f_k(t'_1, \dots, t'_k)$, thus $j = k$, and $t_1 = t'_1, \dots, t_j = t'_k$. Hence it is the same formula.

We have proved the following theorem.

5.19 Theorem. It is provable in $I\Sigma_0 + Exp$ that: a formula has a cut-free proof if and only if it has a Herbrand proof. In symbols

$$I\Sigma_0 + Exp \vdash (\forall x)(CFPr^\bullet(x) \equiv HPr^\bullet(x))\,.$$

By formalizing a classical proof of the consistency of arithmetic we get the following result.

5.20 Theorem.

(i) $I\Sigma_0 + Superexp \vdash Con_{I\Sigma_0}$.

(ii) $I\Sigma_0 + Exp \vdash HCon^\bullet_Q$ and for every k, $I\Sigma_0 + Exp \vdash RCon^\bullet_Q(\overline{k})$.

Proof. First we transform Q into an open theory Q': we delete axiom (Q8) defining $\leq$, add the predecessor function and replace axiom (Q3) by an open

axiom. Assume $\neg HCon(Q')$, hence we have some propositional tautology $He(\neg Q')$, which is a disjunction of substitution instances of $\neg Q'$. But this is impossible, since in $I\Sigma_0 + Exp$ we can define the truth for open formulae. Thus we have $I\Sigma_0 + Exp \vdash HCon^{\bullet}_{Q'}$.

To prove (i), first reduce the consistency of $I\Sigma_0$ to the consistency of Q' using the interpretations of $I\Sigma_0$ in Q (Theorem 5.12) and Q in Q'. Then apply cut-elimination in $I\Sigma_0 + Superexp$ (Theorem 5.17 (i)), and the equivalence of the cut-free and Herbrand consistency (Theorem 5.19).

$RCon^{\bullet}_{Q}(\overline{k})$ is reduced to $RCon^{\bullet}_{Q'}(\overline{m})$, for some m, by the interpretation of Q in Q'. Applying cut elimination (Theorem 5.17 (ii)), this is further reduced to $CFCon^{\bullet}_{Q'}$ and by Theorem 5.19 to $HCon^{\bullet}_{Q'}$. Thus we get

$$I\Sigma_0 + Exp \vdash RCon^{\bullet}_{Q}(\overline{k}).$$

Now $RCon^{\bullet}_{Q}(\overline{0})$ is just $CFCon^{\bullet}_{Q}$ which is equivalent to $HCon^{\bullet}_{Q}$ in $I\Sigma_0 + Exp$. This shows the rest of (ii). □

(e) The Π_1 Theorems of $I\Sigma_0 + Exp$

In this subsection we prove a theorem of A. Wilkie which characterizes Π_1 sentences provable in $I\Sigma_0 + Exp$ as the Π_1 sentences interpretable in Q and consider some related problems. We start with some lemmas. Recall that x denotes the formalized numeral function and that we are using *dyadic* numerals of Sect. 3.

5.21 Lemma. For every inductive formula $I(x)$ in Q, there exists k such that

$$S^1_2 \vdash (\forall x) RPr^{\bullet}_{Q}(\overline{k}, I(x)).$$

Proof. Recall that a numeral $\overline{n}$, for $n > 0$, is constructed as follows. We take $n_0, \ldots, n_k$ such that $n_0 = 1, n_k = n$ and $n_{i-1} = \llcorner n_i/2 \lrcorner$. Then $\overline{n}_0$ is $\overline{1}$ and, for $i = 1, \ldots, k$, $\overline{n}_i$ is

$$\begin{aligned} &\overline{2} * \overline{n}_{i-1}, \quad \text{if } n_i = 2n_{i-1}; \\ &(\overline{2} * \overline{n}_{i-1}) + \overline{1}, \quad \text{if } n_i = 2n_{i-1} + 1; \end{aligned}$$

where $\overline{1}$ is $S(\overline{0})$ and $\overline{2}$ is $(S(\overline{0}) + S(\overline{0}))$. For a given inductive $I(x)$, take some $J(x)$ which is a subcut of I and $J(x) \to J(\overline{2} * x)$, provably in Q. Then to show $J(\overline{n})$ we need at most $2k + 1$ proof steps. The formulae that we use are of polynomial size in the length of n, hence the whole proof has polynomial size. Therefore S^1_2 can formalize this proof. Note that the rank of formulae used in the proof does not depend on $\overline{n}$, hence is determined only by I. □

5.22 Lemma. There exists k such that for every two terms $t(x_1,\dots,x_n)$ and $s(x_1,\dots,x_n)$ in L_0:

$$S_2^1 \vdash (\forall x_1,\dots,x_n)((t(x_1,\dots,x_n) = s(x_1,\dots,x_n) \rightarrow RPr^{\bullet}_Q(\overline{k}, t(\dot{x}_1,\dots,\dot{x}_n) = s(\dot{x}_1,\dots,\dot{x}_n))) \\ \& (t(x_1,\dots,x_n) \neq s(x_1,\dots,x_n) \rightarrow RPr^{\bullet}_Q(\overline{k}, t(\dot{x}_1,\dots,\dot{x}_n) \neq s(\dot{x}_1,\dots,\dot{x}_n)))).$$

In the following proof and the next lemma we shall denote the function assigning the dyadic numeral to a number also by $numeral(x)$; the dot notation would be rather confusing in some formulae.

Proof. Let $I(x)$ be a cut in Q on which the operations $+$ and $*$ have the properties of ring operations. We can use this cut to construct proofs of instances of associativity, commutativity, etc. for numerals. Namely, by the lemma above, we first show that the numerals in question belong to I and then we apply the law in I. This cut will be the only factor that influences the cut-rank k of the resulting proof. Now the sentence

$$RPr^{\bullet}_Q(\overline{k}, t(\dot{x}_1,\dots,\dot{x}_n) = s(\dot{x}_1,\dots,\dot{x}_n))$$

can be easily proved in S_2^1 by induction. We only have to transform it into a Σ_1^b formula, i.e. we have to find a polynomial bound to the length of the proof of $t(\dot{x}_1,\dots,\dot{x}_n) = s(\dot{x}_1,\dots,\dot{x}_n)$. By Lemma 5.21 the length of a proof of $I(\dot{x})$ is polynomial in the length of x. Next we find polynomial proofs of

(e.1) $S(numeral(x_1)) = numeral(S(x_1))$,

(e.2) $numeral(x_1) + numeral(x_2) = numeral(x_1 + x_2)$,

(e.3) $numeral(x_1) * numeral(x_2) = numeral(x_1 * x_2)$.

Perhaps the best way to see a polynomial bound to these statements is the following. First construct proofs of $\dot{x}_i = t_i$ where t_i are terms of the form

$$\overline{2}^{j_1} + \dots + \overline{2}^{j_m}, \quad (j_1 > j_2 > \dots > j_m),$$

where $\overline{2}^k$ denotes $\underbrace{\overline{2} * (\overline{2} * \dots * \overline{2})}_{k \text{ times}}$. Think of t_i as a modified numeral. Then follow the standard algorithms for $+$ and $*$ to obtain the proofs of (e.1–3) for the *modified numerals*. Finally transform the modified numerals back into original $numeral(S(x_1))$, $numeral(x_1+x_2)$ and $numeral(x_1*x_2)$ respectively. As is easily shown by induction, the maximal of these three bounds (the

bound for the length of the proof of (e.3) is asymptotically maximal) is a bound to the proof of the equality of general terms. To prove the inequality, take a suitable term u, prove

$$t(\dot{x}_1, \ldots, \dot{x}_n) + S(u) = s(\dot{x}_1, \ldots, \dot{x}_n),$$

or

$$s(\dot{x}_1, \ldots, \dot{x}_n) + S(u) = t(\dot{x}_1, \ldots, \dot{x}_n),$$

and use the fact that

$$Q \vdash I(x) \,\&\, I(y) \rightarrow x + S(y) \neq x. \qquad \square$$

5.23 Lemma. There exists k such that

$$S_2^1 \vdash (\forall y) RPr_Q^{\bullet}(\overline{k}, (\forall x)(x \leq \mathit{numeral}^{\bullet}(y+1) \equiv (x \leq \mathit{numeral}^{\bullet}(y) \vee x = \mathit{numeral}^{\bullet}(y+1)))).$$

Proof. We construct a cut I in Q such that

$$Q \vdash (\forall x, z)(I(z) \rightarrow (x \leq S(z) \equiv (x \leq z \vee x = S(z)))). \tag{e.4}$$

First take J defined by

$$J(x) \equiv_{df} 0 + x = x \,\&\, (\forall z)(S(z) + x = S(z + x)).$$

One can easily show that J is inductive. Now take I to be a *subcut* of J and verify (e.4). We shall work in S_2^1. Let a number y be given. By Lemma 5.21 we have $Q \vdash I(\mathit{numeral}(y))$, hence by (e.4)

$$Q \vdash x \leq S(\mathit{numeral}(y)) \equiv (x \leq \mathit{numeral}(y) \vee x = S(\mathit{numeral}(y))).$$

By Lemma 5.22 $Q \vdash \mathit{numeral}(y+1) = S(\mathit{numeral}(y))$. Thus Q proves

$$(\forall x)(x \leq \mathit{numeral}(y+1) \equiv (x \leq \mathit{numeral}(y) \vee x = \mathit{numeral}(y+1))).$$

All that we have used were proofs of bounded cut-rank. $\square$

For the following lemma, we shall fix some translation of L_2 into L_0 which determines an interpretation of *BASIC* on some cut in Q. Such a translation exists, since $I\Sigma_0 + \Omega_1$ is interpretable in Q by Theorem 5.12. We shall say that a formula ψ of L_0 is *essentially* Σ_1^b (resp. Π_1^b), if it is the translation of a Σ_1^b (resp. Π_1^b) formula of L_2.

5.24 Lemma (formalized Σ completeness of Q).

(i) Let $\psi(x,y)$ be a bounded formula in L_0. Then there exists k such that

$$I\Sigma_0 + Exp \vdash (\forall x)((\exists y)\psi(x,y) \to RPr^{\bullet}_{Q}(\overline{k}, (\exists y)\psi(\dot{x},y)))\,.$$

(ii) Let $\psi(x,y)$ be essentially Σ^b_1. Then there exists k such that

$$S^1_2 \vdash (\forall x)((\exists y)\psi(x,y) \to RPr^{\bullet}_{Q}(\overline{k}, (\exists y)\psi(\dot{x},y)))\,.$$

Proof. First we observe that in both cases it suffices to prove only

$$(\forall x)(\psi(x,y) \to RPr^{\bullet}_{Q}(\overline{k}, \psi(\dot{x},\dot{y})))\,,$$

since $(\exists y)\psi(\dot{x},y)$ will follow by an application of the rule $(\exists)$.

(i) The task of proving $\psi(\dot{x},\dot{y})$ is similar to the task of deciding its truth: we shall prove $\psi(\dot{x},\dot{y})$ by proving numerical instances of the open part of ψ. Thus we can bound the size of the numerals needed in this proof in the same way as in the proof of Theorem 5.4. The numerical instances of the open part of ψ follow from the numerical instances of atomic and negated atomic subformulae. The existence of bounded cut-rank proofs of atomic sentences with $=$ and their negations follows from Lemma 5.22. The atomic sentences with $\leq$ and their negations are reduced to atomic sentences with $=$ in the same way as the negations of atomic sentences with $=$ had been.

We must be a little careful when pasting the proofs of the numerical instances into a proof of a bounded sentence, since we need a proof of *bounded cut-rank*. A sentence of the form $(\exists y \leq t)\alpha(y)$ follows from some $\alpha(\overline{n})$, where n is less than or equal to the value of t. A proof of $(\forall y \leq t)\alpha(y)$ is constructed from the proofs of $\alpha(\overline{0}), \alpha(\overline{1}), \ldots, \alpha(\overline{m})$, where m is the value of t, as follows. We prove successively $(\forall y \leq \overline{n})\alpha(y)$, for $n = 0, \ldots, m$. For $n = 0$ it is an immediate consequence of $\alpha(\overline{0})$; to obtain it for $n+1$ from a proof for n, use Lemma 5.23 and $\alpha(\overline{n+1})$. Finally use Lemma 5.22 to show $Q \vdash (\forall y \leq t)\alpha(y) \equiv (\forall y \leq \overline{m})\alpha(y)$.

(ii) The proof of (ii) is only a modification of the proof of (i) where we replace *bounded* by *sharply bounded*. We also need an extension of Lemma 5.23 to L_2, instead of L_0, and $BASIC$, instead of Q. Again the proof is essentially the same: translate the natural polynomial time algorithms for the evaluation of terms in L_2 into a proof using equalities which hold on a suitable cut in $BASIC$. We omit the details. □

5.25 Lemma. Let $\psi(x)$ be a bounded formula and suppose

$$I\Sigma_0 + Exp \vdash (\forall x)\psi(x)\,.$$

Then there exists k such that

$$I\Sigma_0 \vdash (\forall x)((\exists y)(y = 2_k^x) \to \psi(x)) .$$

Proof. Suppose $I\Sigma_0$ does not prove $(\forall x)((\exists y)(y = 2_k^x) \to \psi(x))$, for any k. Then, by compactness, $I\Sigma_0 + \{(\exists y)(y = 2_k^c) \& \neg\psi(c) \mid k \in N\}$ (c a new constant) is consistent. Take a model M of this theory and an initial segment of it

$$K = \{a \in M \mid (\exists k \in N)(M \vDash a \leq 2_k^c)\} .$$

Then K is a model of $I\Sigma_0 + Exp$, but $K \vDash \neg\psi(c)$, because ψ is bounded. □

Now we are ready for the theorems.

5.26 Theorem. Let $\psi(x)$ be a bounded formula. Then the following are equivalent:

(1) There exists a cut $I(x)$ in Q such that

$$Q \vdash (\forall x)(I(x) \to \psi(x)) .$$

(2) There exists an interpretation of $Q + (\forall x)\psi(x)$ in Q.
(3) $I\Sigma_0 + Exp \vdash (\forall x)\psi(x)$.

Proof. (1) $\Rightarrow$ (2) follows from the existence of shortenings of cuts closed under $+$ and $*$.

(2) $\Rightarrow$ (1) Let ι be an interpretation of $Q + (\forall x)\psi(x)$ in Q. We shall construct a cut $I(x)$ in $I\Sigma_0$ such that

$$I\Sigma_0 \vdash (\forall x)(I(x) \to \psi(x)) .$$

This is sufficient, since $I\Sigma_0$ has an interpretation in Q determined by a cut. The cut $I(x)$ is defined by the formalization of the following formula: "There exists an isomorphism of the initial segment $[\overline{0}, x]$ onto an initial segment in the sense of interpretation ι". Here we take intervals as structures with $=$, $\leq$, and $+$, $*$ as ternary relations. An isomorphism is a number which codes a sequence of pairs. We leave it to the reader to write down an explicit formula for I and check that it is a cut in $I\Sigma_0$. Let us only note that the extension of an isomorphism with domain $[\overline{0}, x]$ to an isomorphism with domain $[\overline{0}, S(x)]$ is obtained by adding to it a pair $(S(x), z)$, where z is the ι successor of the image of x. Such an extension of a sequence is possible in $I\Sigma_0$ (but not in Q). Now the fact that $(\forall x)\psi(x)$ holds in the sense of ι is transferred by the isomorphisms onto I.

(1) $\Rightarrow$ (3) Suppose

$$Q \vdash (\forall x)(I(x) \to \psi(x)) .$$

By Lemma 5.21

$$I\Sigma_0 + Exp \vdash (\forall x) RPr^{\bullet}_Q(\overline{k}, I(x)),$$

for some constant k. Hence

$$I\Sigma_0 + Exp \vdash (\forall x) RPr^{\bullet}_Q(\overline{k}, \psi(x)).$$

By Theorem 5.20, $I\Sigma_0 + Exp$ proves the consistency of Q with respect to proofs of cut-rank k. Thus we have

$$I\Sigma_0 + Exp \vdash (\forall x) \neg RPr^{\bullet}_Q(\overline{k}, \neg\psi(\dot{x})).$$

By Lemma 5.24(i)

$$I\Sigma_0 + Exp \vdash (\forall x)(\neg\psi(x) \to RPr_Q(\overline{m}, \neg\psi(\dot{x})))$$

for some m. We can assume that $k = m$ (otherwise take $\max(k, m)$ instead of k and m). Hence (3) follows from the last two statements.

(3) $\Rightarrow$ (1) Suppose $I\Sigma_0 + Exp \vdash (\forall x)\psi(x)$. By the lemma above we have some k such that

$$I\Sigma_0 \vdash (\forall x)((\exists y)(y = 2^x_k) \to \psi(x)).$$

Take cuts I, J in Q such that J determines an interpretation of $I\Sigma_0$ in Q and I is a subcut of J such that

$$Q \vdash I(x) \to (\exists y)(y = 2^x_k \,\&\, J(y)).$$

Then we have $Q \vdash (\forall x)(I(x) \to \psi(x))$. □

5.27 Theorem. Let $\psi(x)$ be a bounded formula in L_0. Then the following are equivalent:

(1) $I\Sigma_0 + Exp \vdash (\forall x)\psi(x)$.

(2) For some k

$$S^1_2 \vdash (\forall x) RPr^{\bullet}_Q(\overline{k}, \psi(\dot{x})).$$

If $\psi(x)$ is the translation of a Π^b_1 formula $\varphi(x)$ into L_0, then (1) and (2) are equivalent to:

(3) For some k

$$S^1_2 + RCon^{\bullet}_Q(\overline{k}) \vdash (\forall x)\varphi(x).$$

Proof. (1) $\Rightarrow$ (2) Suppose $I\Sigma_0 + Exp \vdash (\forall x)\psi(x)$, ψ bounded. Then by Theorem 5.26 we have $Q \vdash (\forall x)(I(x) \to \psi(x))$ for some cut I in Q. By Σ_1 completeness

$$S^1_2 \vdash RPr^{\bullet}_Q(\overline{j}, (\forall x)(I(x) \to \psi(x))),$$

for some j. By Lemma 5.21

$$S_2^1 \vdash (\forall x) RPr_Q^\bullet(\overline{m}, I(\dot{x}))$$

for some m. Take k to be the maximum of j and m, then we have (2).

(2) $\Rightarrow$ (1) By definition

$$\text{(e.5)} \qquad S_2^1 + RCon_Q^\bullet(\overline{k}) \vdash (\forall x) RPr_Q^\bullet(\overline{k}, \psi(\dot{x})) \to (\forall x) \neg RPr_Q^\bullet(\overline{k}, \neg\psi(\dot{x})) .$$

By Theorem 5.20, $RCon_Q^\bullet(\overline{k})$ is provable in $I\Sigma_0 + Exp$, thus (e.5) is provable in $I\Sigma_0 + Exp$. Since we assume (2), we have

$$I\Sigma_0 + Exp \vdash (\forall x) \neg RPr_Q^\bullet(\overline{k}, \neg\psi(\dot{x})) .$$

By Lemma 5.24(i)

$$I\Sigma_0 + Exp \vdash (\forall x)(\neg RPr_Q^\bullet(\overline{k}, \neg\psi(\dot{x})) \to \psi(x)) .$$

Hence (2) implies (1).

Now assume that ψ is the translation of a Π_1^b formula $\varphi(x)$ into L_0.

(2) $\Rightarrow$ (3) By Lemma 5.24(ii) we have

$$S_2^1 \vdash (\forall x)(\neg RPr_Q^\bullet(\overline{k}, \neg\psi(\dot{x})) \to \psi(x)) .$$

Then using (e.5) we get

$$S_2^1 + RCon_Q^\bullet(\overline{k}) \vdash (\forall x)\psi(x) .$$

from (2). But $\varphi(x)$ is equivalent to $\psi(x)$ in S_2^1, thus we have (3).

(3) $\Rightarrow$ (1) This is because $I\Sigma_0 + Exp$ "contains" S_2^1 and proves $RCon_Q^\bullet(\overline{k})$. □

We would like to axiomatize Π_1 consequences of $I\Sigma_0 + Exp$ over, say, S_2^1 (put otherwise, we want to find some nice basis). The equivalence of (1) with (3) gives only a partial answer: since $RCon_Q^\bullet(\overline{k})$ can be written as a formula of the form $(\forall x)\psi(x)$ with ψ in Π_1^b, the set of sentences $RCon_Q^\bullet(\overline{k})$, $k = 0, 1, \ldots$ axiomatizes the $\forall\Pi_1^b$ consequences of $I\Sigma_0 + Exp$.

(f) Incompleteness Theorems

It would be paradoxical if the incompleteness theorems did not hold in weak systems. Of course formalization of syntax in weaker theories is a more difficult task, therefore we cannot apply classical proofs of the second Gödel incompleteness theorem quite directly. It turns out that the second

incompleteness theorem can be even strengthened. A typical example is the unprovability of the consistency of Q in $I\Sigma_0 + Exp$. The results presented in this subsection will be corollaries of the theorems proved above and of the following theorem.

Note that in the following theorem (i) holds for every finitely axiomatized sequential theory containing Q. We shall not prove this general theorem here. For finite theories in the language of arithmetic extending $I\Sigma_1$, statement (i) follows from Theorem 3.20(1) Chap. III. Statement (ii) generalizes Theorem 3.11, Chap. III, to weaker theories; however the assumption about the axiomatizability is stronger here. It is possible to use here the same assumption about the axiomatizability as in Chap. III, but we shall not prove this strengthening. Recall that $RCon_T^{\bullet I}(\overline{k})$ is

$$(\forall z)(I(z) \,\&\, c\text{-}rank(z) \leq \overline{k} \to \neg Proof_T^{\bullet}(z, \overline{0=1}))\,.$$

5.28 Theorem. (i) For every k there exists a cut I in Q such that

$$Q \vdash RCon_Q^{\bullet I}(\overline{k})\,;$$

(ii) Let T be a consistent theory containing Q and having a Σ_1^b axiomatization. Then for every cut J in T there exists m such that

$$\neg(T \vdash RCon_T^{\bullet J}(\overline{m}))\,.$$

Proof. (i) By Theorem 5.20(ii), for every k

$$I\Sigma_0 + Exp \vdash RCon_Q^{\bullet}(\overline{k})\,.$$

By Theorem 5.26, this implies the first statement.

(ii) The proof of this statement will be similar to the proof of Theorem 3.11, Chap. III. Roughly speaking, we only have to take m sufficiently large. Since this result plays a key role in the rest of this section, we shall prove it in detail.

Because of the shortening techniques, we may suppose that J has some additional properties. Thus we shall assume that J is a cut in Q and it determines an interpretation of S_2^1 in Q. In particular, it is provable in Q, hence also in T, that for any two sequences in J, their concatenation lies also in J.

We would like to use the provability conditions of 2.16, Chap. III, for $RPr_T(\overline{m}, x)$. We shall prove the first two. The third one, which is formalized *Modus Ponens*, is clearly false, since *Modus Ponens* increases the cut-rank. Thus we shall first take a suitable m, and then we shall follow the usual proof

of the second incompleteness theorem and check that the third condition is actually applied only to formulae of rank at most m. We shall divide the proof into several claims.

Claim 1. For every sentence φ, if $T \vdash \varphi$, then $T \vdash CFPr_T^{\bullet J}(\overline{\varphi})$.

Proof. Suppose $T \vdash \varphi$. By cut-elimination we have a cut-free proof of φ in T. By the Σ-completeness of Q, $Q \vdash CFPr_T^{\bullet}(\overline{\varphi})$. Since J determines an interpretation of Q in T, we get the conclusion of the claim. □

Claim 2. There exists k such that

$$T \vdash RPr_T^{\bullet J}(x,y) \to RPr_T^{\bullet J}(\overline{k}, \overline{RPr_T^{\bullet J}(\dot{x},\dot{y})}).$$

Proof. Let us first write down $RPr_T^{\bullet J}(x,y)$ more explicitly:

$$(\exists d)(J(d) \,\&\, J(x) \,\&\, J(y) \,\&\, c\text{-}rank(d) \leq x \,\&\, Proof_T(d,y)).$$

By Lemma 5.24, for some k_1

$$S_2^1 \vdash rank(d) \leq x \,\&\, Proof_T^{\bullet}(d,y) \to \\ \to RPr_Q^{\bullet}(\overline{k}_1, \overline{rank(\dot{d}) \leq \dot{x} \,\&\, Proof_T^{\bullet}(\dot{d},\dot{y})}).$$

By Lemma 5.21, for some k_2

$$S_2^1 \vdash RPr_Q(\overline{k}_2, \overline{J(\dot{d}) \,\&\, J(\dot{x}) \,\&\, J(\dot{y})}).$$

To obtain $RPr_T^{\bullet J}(\dot{x},\dot{y})$ we need only the rules for & and $\exists$ which do not increase the cut-rank. Hence for $k = \max(k_1, k_2)$

$$S_2^1 \vdash RPr_T^{\bullet}(x,y) \to RPr_Q^{\bullet}(\overline{k}, \overline{RPr_T^{\bullet J}(\dot{x},\dot{y})}).$$

Here we can replace Q by T, since Q is contained in T. As $T \vdash (S_2^1)^J$, we get

$$T \vdash (RPr_T^{\bullet}(x,y) \to RPr_T^{\bullet}(k, \overline{RPr_T^{\bullet J}(\dot{x},\dot{y})}))^J,$$

which is in fact the statement of the claim. □

By 2.1, Chap. III, we have some $\delta(x)$ such that

$$T \vdash \delta(x) \equiv \neg RPr_T^{\bullet J}(x, \overline{\delta(\dot{x})}).$$

$\overline{\delta(\dot{x})}$ is the numeral of the formula $\delta(numeral(x))$. Now we can choose m. Let m be such that

(1) $m \geq k$, where k is from Claim 2;
(2) $m \geq rank(RPr_T^{\bullet J}(x, y))$;
(3) $m \geq rank(\delta(x))$.

Note that we thus have also $m \geq rank(RPr_T^{\bullet J}(\overline{m}, y))$ and $m \geq rank(\delta(\overline{m}))$. Let us define also

$$\Delta =_{df} \delta(\overline{m});$$
$$\alpha =_{df} RPr_T^{\bullet J}(\overline{m}, \overline{\Delta}).$$

Claim 3. $T \vdash \Delta \equiv \neg RPr_T^{\bullet J}(\overline{m}, \overline{\Delta})$.

Proof. By definition we have

$$T \vdash \delta(\overline{m}) \equiv \neg RPr_T^{\bullet J}(\overline{m}, \overline{\delta(\dot{\overline{m}})}).$$

Thus we only need

$$T \vdash \overline{\delta(\dot{\overline{m}})} = \overline{\delta(\overline{m})},$$

which follows from the Σ-completeness of Q, hence of T. □

Claim 4. $\neg(T \vdash \Delta)$.

Proof. Suppose $T \vdash \Delta$. Then, by Claim 1, $T \vdash CFPr_T^{\bullet J}(\overline{\Delta})$, whence $T \vdash RPr_T^{\bullet J}(\overline{m}, \overline{\Delta})$. By Claim 3 this means that T is inconsistent, which is a contradiction. □

Claim 5. $T \vdash RPr_T^{\bullet J}(\overline{m}, \overline{\alpha}) \,\&\, RPr_T^{\bullet J}(\overline{m}, \overline{\neg\alpha}) \rightarrow \neg RCon_T^{\bullet J}(\overline{m})$.

Proof. If α and $\neg\alpha$ are derivable using proofs of cut-rank of at most m, then so are $\alpha \vee \overline{0} = \overline{1}$ and $\neg\alpha \vee \overline{0} = \overline{1}$ (add $\overline{0} = \overline{1}$ to each proof line as a side formula). Since $m \geq rank(RPr_T^{\bullet J}(x, y)) = rank(\alpha)$, we get a proof of $\overline{0} = \overline{1}$ by taking the concatenation of these proofs and applying the cut rule. The closure properties of J ensure that such a proof will be also in J. □

Claim 6. $T \vdash RPr_T^{\bullet J}(\overline{m}, \overline{\Delta}) \rightarrow RPr_T^{\bullet J}(\overline{m}, \overline{\alpha})$.

Proof. By Claim 2

$$T \vdash RPr_T^{\bullet J}(\overline{m}, \overline{\Delta}) \rightarrow RPr_T^{\bullet J}(\overline{m}, \overline{RPr_T^{\bullet J}(\dot{\overline{m}}, \dot{\overline{\Delta}})}).$$

By the definition of α and Σ-completeness

$$T \vdash \overline{RPr_T^{\bullet J}(\dot{\overline{m}}, \dot{\overline{\Delta}})} = \overline{\alpha}.$$

□

Claim 7. $T \vdash RPr_T^{\bullet J}(\overline{m}, \overline{\Delta}) \to RPr_T^{\bullet J}(\overline{m}, \overline{\neg\alpha})$.

Proof. By Claim 3 and the definition of α, $T \vdash \Delta \to \neg\alpha$. By Claim 1, this implies $T \vdash RPr_T^{\bullet J}(\overline{m}, \overline{\Delta \to \neg\alpha})$. If we have proofs of Δ and $\Delta \to \neg\alpha$ of cut-rank m, we can easily combine them into a proof of cut-rank m of $\neg\alpha$, because $m \geq rank(\Delta)$. Again, the closure properties of J ensure that this proof is in J too. □

Now we can finish the proof of part (ii) of the theorem. By Claims 3, 6, 7 and 5, we have in T

$$\neg\Delta \to RPr_T^{\bullet J}(\overline{m}, \overline{\Delta}) \to (RPr_T^{\bullet J}(\overline{m}, \overline{\alpha}) \,\&\, RPr_T^{\bullet J}(\overline{m}, \overline{\neg\alpha})) \to \neg RCon_T^{\bullet J}(\overline{m}),$$

i.e. $T \vdash RCon_T^{\bullet J}(\overline{m}) \to \Delta$. Hence, by Claim 4, $RCon_T^{\bullet J}(\overline{m})$ is not provable in T. □

5.29 Corollary. Con_Q is not provable in $I\Sigma_0 + Exp$.

Proof. By the theorem above, Q does not prove $Con_Q^{\bullet J}$ for any cut I. Hence by Theorem 5.26 $I\Sigma_0 + Exp$ does not prove $Con_Q^{\bullet}$. □

5.30 Corollary. Cut elimination (unrestricted), and Herbrand's Theorem are not provable in $I\Sigma_0 + Exp$.

Proof. By Theorem 5.20(ii) $I\Sigma_0 + Exp$ proves cut-free and Herbrand consistency of Q. If we had cut-elimination or Herbrand's Theorem in $I\Sigma_0 + Exp$, we could reduce the ordinary consistency of Q to the restricted one. □

5.31 Theorem. $I\Sigma_0 + Exp + Con_{I\Sigma_0}^{\bullet}$ does not prove $Con_{I\Sigma_0+Exp}^{\bullet}$.

Proof. Let J be the cut in $I\Sigma_0 + Exp$ defined by

$$J(x) \equiv_{df} (\exists y)(y = 2_x^0).$$

We shall show that

(f.1) $$I\Sigma_0 + Exp \vdash Con_Q^{\bullet J}.$$

Suppose not. Let us work in $I\Sigma_0 + Exp$, let d be a proof of a contradiction in J. Then d has cut-rank at most $lh(d)$. We have

$$2_{3*lh(d)}^{lh(d)} \leq 2_d^0,$$

since d must be nonstandard and d is exponentially larger than $lh(d)$; hence such a number exists. Thus we can apply cut elimination to obtain a cut-free

proof of a contradiction in Q (apply Lemma 5.15 $lh(d)$-times and use the inequality

$$2^{2^{2*lh(d)}} \leq 2^{2^{2^{lh(d)}}}).$$

But this is not possible by 5.20 (ii).

Now suppose the statement of the theorem is false. Thus

$$I\Sigma_0 + Exp \vdash Con^{\bullet}_{I\Sigma_0} \rightarrow Con^{\bullet}_{I\Sigma_0+Exp}.$$

Since $Con_{I\Sigma_0}$ and Con_Q are equivalent in $I\Sigma_0 + Exp$, we have

$$I\Sigma_0 + Exp \vdash Con^{\bullet}_{Q} \rightarrow Con^{\bullet}_{I\Sigma_0+Exp}.$$

Using an obvious Lemma 5.25, we can deduce that provably in $I\Sigma_0 + Exp$, a proof d of a contradiction in $I\Sigma_0 + Exp$ can be transformed into a proof d' of a contradiction in Q such that $d' \leq 2^d_k$ for a suitable standard k. Let I be a subcut of J such that

$$I\Sigma_0 + Exp \vdash I(x) \rightarrow J(2^x_{\overline{k}}).$$

Hence a proof d of a contradiction in $I\Sigma_0 + Exp$ such that d is in I can be transformed into a proof d' of a contradiction in Q such that d' is in J. Thus, by (f.1) no such proof can exist, i.e. $I\Sigma_0 + Exp$ proves $Con^I_{I\Sigma_0+Exp}$. But this is in contradiction to Theorem 5.28 (ii). □

5.32 Corollary. $I\Sigma_0 + Exp$ is not interpretable in $I\Sigma_0$.

Proof. Suppose $I\Sigma_0 + Exp$ is interpretable in $I\Sigma_0$. It is well-known that if T is interpretable in S and S is consistent, then T is consistent too. This simple theorem can be easily formalized in $I\Sigma_0 + Exp$, if T is finitely axiomatized. Since $I\Sigma_0 + Exp$ is finitely axiomatizable (Theorem 5.6), as one can check provably in $I\Sigma_0 + Exp$, we get

$$I\Sigma_0 + Exp \vdash Con^{\bullet}_{I\Sigma_0} \rightarrow Con^{\bullet}_{I\Sigma_0+Exp},$$

which contradicts Theorem 5.31. □

5.33 Theorem. For every k there exists m such that $RCon^{\bullet}_Q(\overline{m})$ is not provable in $I\Sigma_0 + \Omega_n + RCon^{\bullet}_Q(\overline{k})$ for any n.

Proof. We shall use Theorem 5.28 again. Let I be a cut in Q such that

$$Q \vdash RCon^{\bullet I}_Q(\overline{k}).$$

We can suppose that I determines an interpretation of $I\Sigma_0$. Let J be a subcut of I such that

$$Q \vdash J(x) \to I(2^{2x}).$$

We know that, for some m, $Q + \neg RCon_Q^{\bullet J}(\overline{m})$ is consistent. Let M be a model of this theory. Let K be its initial segment defined by

$$K = \{a \in M; (\exists b \in J^M)(\exists i \in N)(a \leq \omega_i(b))\}.$$

First we shall show that K is closed under all functions ω_n. Let $a \in K$; without loss of generality we can suppose that a is nonstandard. Then, for some nonstandard $b \in J^M$, $a \leq \omega_i(b)$. Let n be given. Then

$$\begin{aligned}\omega_n(a) &\leq \omega_n(\omega_i(b)) \leq \omega_{\max(n,i)}(\omega_{\max(n,i)}(b)) \\ &\leq \omega_{\max(n,i)+1}(b) \in K.\end{aligned}$$

To prove the last inequality, use induction on j to show that

$$\omega_j(\omega_j(x)) \leq \omega_{j+1}(x),$$

for every sufficiently large x (hence for every nonstandard x). Now we need that $K \subseteq I^M$. This follows from the definition of the cuts and from the following inequality, which is also easily provable by induction on j:

$$\omega_j(x) \leq 2^{2x}, \quad \text{for } x \text{ sufficiently large}.$$

Since K is contained in I^M, which is a model of $I\Sigma_0$, K is a model of $I\Sigma_0 + \omega_n$, for every n. Since $Q \vdash RCon_Q^{\bullet J}(\overline{k})$ and $M \models \neg RCon_Q^{\bullet J}(\overline{m})$, we have

$$K \models RCon_Q^{\bullet}(\overline{k}) \,\&\, \neg RCon_Q^{\bullet}(\overline{m}),$$

which finishes the proof. □

This theorem gives us another piece of important information about $\forall\Pi_1^b$ consequences of $I\Sigma_0 + Exp$: they are not "finitely based". Further we have this consequence.

5.34 Corollary. $I\Sigma_0 + Exp$ is not Π_1 conservative over $I\Sigma_0 + \{\Omega_n;\ n \in N\}$.

In fact this is all that we know about Π_1 conservativity between systems of bounded arithmetic. In particular it seems plausible that $I\Sigma_0 + \Omega_1$ is not Π_1 conservative over $I\Sigma_0$, but we cannot even prove that $I\Sigma_0 + \{\Omega_n;\ n \in N\}$ is not Π_1 conservative over $I\Sigma_0$. A Π_1 sentence which might separate $I\Sigma_0 + \Omega_1$ from $I\Sigma_0$ is the so-called Bertrand's postulate. It is the theorem (due to Sylvester) which says that there is a prime between n and $2n$, for every n.

This sentence is provable in $I\Sigma_0 + \Omega_1$, but no proof of it has been found in $I\Sigma_0$. Whether such a proof exists is an important open problem.

(g) On the Limited Use of Exponentiation

If we are interested only in Π_1 sentences, then it seems that all such sentences which are also truly mathematical results are provable in $I\Sigma_0 + \mathit{Exp}$ (we consider sentences like $\mathit{Con}(Q)$ to be uninteresting for an ordinary mathematician). In fact the proofs use exponentiation at most once. This means that for instance if we construct some graph, we may consider the set of all graphs of a given size, but usually we do not consider the set of subsets, etc. Put otherwise, we use the finite power set operation only once. This phenomenon is worthwhile to formalize, but it cannot be done directly in first order logic. In first order theory we have either to accept axiom Exp, and then we have also all finite iterations of 2^x, or we do not have exponentiation at all. A possible approach is to use higher order bounded arithmetics. We shall use a different approach. Our aim is, roughly speaking, to show that there is an infinite hierarchy according to the number of applications of the exponentiation axiom.

So what does it really mean that $(\forall x)\varphi(x)$ is provable using exponentiation only once? This means that we prove $\varphi(x)$ assuming that 2^x exists; formally

$$I\Sigma_0 + \Omega_1 \vdash (\forall x)((\exists y)(y = 2^x) \to \varphi(x)) .$$

It is convenient to consider $I\Sigma_0 + \Omega_1$ instead of $I\Sigma_0$, since thus we obtain a concept which is more robust: in $I\Sigma_0 + \Omega_1$, 2^x exists iff 2^{x^2} exists, etc. There is an equivalent and shorter way to express that $(\forall x)\varphi(x)$ is provable using exponentiation only once, which is

$$I\Sigma_0 + \Omega_1 \vdash (\forall x)\varphi(|x|) .$$

Similarly, k applications correspond to the provability of $(\forall x)\varphi(|x|^{(k)})$, where $|x|^{(k)}$ denotes the k times iterated function $x \mapsto |x|$.

Finally let us also mention a model theoretical characterization of provability using exponentiation only once. Let $I(x)$ be defined by

$$I(x) \equiv_{df} (\exists y)(y = 2^x) .$$

Let K be the class of models of the form

$$K = \{a \in M;\ M \models I\Sigma_0 + \Omega_1 \,\&\, M \models I(a)\} ,$$

with the operations inherited from M. Then the sentences in question are just the sentences true in all models of K.

Let $\psi(x,y)$ be a bounded formula such that for every m

$$I\Sigma_0 + \Omega_1 \vdash (\forall y)\psi(\overline{m},y) \equiv RCon^{\bullet}_Q(\overline{m}).$$

5.35 Lemma. For every k there exists m such that $(\forall y)\psi(\overline{m},|y|^{(k)})$ is not provable in $I\Sigma_0 + \Omega_1$.

Proof. Let $I(x)$ be a cut in Q such that I determines an interpretation of $I\Sigma_0 + \Omega_1$. Let $J(x)$ be a subcut of $I(x)$ such that

$$Q \vdash J(x) \to I(2^x_k).$$

By Theorem 5.28, there exists m such that

$$\neg(Q \vdash RCon^{\bullet J}_T(\overline{m})).$$

Now suppose that $(\forall y)\psi(\overline{m},|y|^{(k)})$ is provable in $I\Sigma_0 + \Omega_1$. Then

$$Q \vdash ((\forall y)\psi(\overline{m},|y|^{(k)}))^I,$$

hence, by the construction of J,

$$Q \vdash ((\forall y)\psi(\overline{m},y))^J,$$

which is a contradiction. □

5.36 Theorem. For every k there exists a bounded formula $\varphi(x)$ such that

$$I\Sigma_0 + \Omega_1 \vdash (\forall x)\varphi(|x|^{(k+1)}),$$

but

$$\neg(I\Sigma_0 + \Omega_1 \vdash (\forall x)\varphi(|x|^{(k)})).$$

Proof. Let m be such that $(\forall y)\psi(\overline{m},|y|^{(k)})$ is not provable in $I\Sigma_0 + \Omega_1$. Since $RCon^{\bullet}_Q(\overline{m})$, i.e. $(\forall y)\psi(\overline{m},y)$, is provable in $I\Sigma_0 + Exp$, it must be provable using exponentiation a finite number of times, see Lemma 5.25. Thus, for some n,

$$I\Sigma_0 + \Omega_1 \vdash (\forall x)\psi(\overline{m},|x|^{(n)}).$$

Hence for some i, $k \leq i < n$,

$$I\Sigma_0 + \Omega_1 \vdash (\forall x)\psi(\overline{m},|x|^{(i+1)}),$$

and

$$\neg(I\Sigma_0 + \Omega_1 \vdash (\forall x)\psi(\overline{m}, |x|^{(i)}))\,.$$

Take $\varphi(x)$ to be $\psi(\overline{m}, |x|^{(i-k)})$, then the theorem follows. □

The instances of the schema $PHP\Sigma_0$ are Π_1 and they are provable using one exponentiation. Actually, most of the proofs that require exponentiation can be reduced to $PHP\Sigma_0$. Still it seems that $PHP\Sigma_0$ is not sufficient to prove all sentences derivable using one exponentiation. We are not able to derive $PHP\Sigma_0$ in $I\Sigma_0 + \Omega_1$, so it is possible that some instances could be used as more natural formulae in the theorem above for $k = 0$. However we do not know which. The point is that the known mathematical statements which can be derived using $PHP\Sigma_0$ can be derived also from a weaker version of PHP which is already provable in $I\Sigma_0 + \Omega_1$ (Bertrand's postulate is an example).

The related problems of whether $I\Sigma_0+\Omega_i$ is Π_1 conservative over $I\Sigma_0+\Omega_j$, for $j < i$, are still open. They are typical in the sense that they are related to open problems in complexity theory. For instance, the question whether $I\Sigma_0 + \Omega_1$ is Π_1 conservative over $I\Sigma_0$ seems to be formally related to the open problem whether $PH = LinH$ (though we are not able *to prove* any relation). New techniques will need to be developed in order to solve such problems. Solving such problems in Bounded Arithmetic might be the first step in solving the corresponding persistent problems in complexity theory.