Errata of *Metamathematics of First-Order Arithmetic*

P.P.

October 16, 2004

We are grateful to all who told us about errors in the book. If you spot errors that are not listed below (there are surely many), please, send them to `pudlak@math.cas.cz`

page 33, line -1. $x + z = y$ should be $y + z = x$

page 34, line 10. $(\exists! u \leq x)(\exists! v < y)(x = yu + v)$ should be $(\exists! u, v)(u \leq x \ \&\ v < y \ \&\ x = yu + v)$

page 139, line -10. $(a, \ldots, a_q)$ should be $(a_1, \ldots, a_q)$

page 140, line 19. $(a_0, \ldots, a_q)$ should be $(a_0, \ldots, a_{q-1})$

page 226, 1.51 Theorem The proof is not correct: $I^{k+1}(K)$ may contain numbers between ω and $K^{k+1}(K) \setminus \omega$, thus we cannot infer from $\prec_{k+2}$ that every element of $I^{k+1}(K)$ is Σ_{k+1}-definable.

Here is a correct proof: We know that $B\Sigma_{k+1}$ is true in $I^{k+1}(K)$ and it fails in $K^{k+1}(K)$. Notice that $B\Sigma_{k+1}$ is a Π_{k+3} formula. Hence an instance of this schema that fails in $K^{k+1}(K)$ and is true in $I^{k+1}(K)$ is Σ_{k+3}. Thus it is not possible that $K^{k+1}(K) \prec_{k+2} I^{k+1}(K)$.

page 254, line -9. the first u should be ν

page 255, 3.30 Lemma. the assumption $v > u$ should be added

page 272, line -9. Π_0^f should be Π_1^f

page 336, line-4. The index at $\{e\}$ should be $i+1$. Formula $\{\bar{e}\}_{i+1}(x) = y$ is ∇_{i+1} and Δ^b_{i+1} only for standard e. Otherwise we do not have a polynomial bound on time. However, we can consider computations that run in time which is not bounded by a standard polynomial if we can prove that a bound exists. If we include the bound as a part of the input for the machine, then we get a standard polynomial bound also in such a case. So the defining formulas must include such a bound.

pages 343-347. There are several errors in the proof of Lemma 4.30 and related results. I do not know how to fix them without introducing more theory. Below I will sketch a possible way based on a class of formulas that formalize Δ^p_{i+1}, predicates decidable in polynomial time using oracles from Σ^p_i. I will start with the theory and then indicate how to fix the errors.

Define formula α to be ∇_{i+1} in a theory T, if there exists a number e such that

$$T \vdash \ \alpha(x_1, \dots, x_n) \ \equiv \ \{\bar{e}\}_{i+1}((x_1, \dots, x_n)) = 0.$$

(I am defining this class only for the proofs in Section 4; it has nothing to do with other possible uses of ∇.)

We need several properties of the class ∇_{i+1} in T^i_2. The main tool for proving these properties, as well as the basic properties of the formalization of $\Box^p_i$ functions (properties (1)-(4) on page 335) is the following. Let us call a pair (w, q) a *quasicomputation* of machine e on input a and for $|b|+1$ steps of computation if the formula $\psi(w, q, a, b)$ on page 333 is satisfied. Recall that it is like a computation with a Σ^b_i oracle, but the computation not always follows the advice of the oracle. Furthermore, we know in T^i_2 that among quasicomputations there is a computation, which is a the unique quasicomputation that follows all advices of the oracle. If we combine several machines into another one we need to prove that the constructed machine does what we intend to do. If we used computations, we would not have enough induction for that unless the number of combined computations were constant. Therefore, we first prove that every quasicomputation is a combination of quasicomputations of the form that we need and then deduce that the same must hold for computations.

I will show it on an example. Let us prove that the class is closed under universal sharply bounded quantification. Suppose $\alpha(x, y)$ is ∇_{i+1} in T^i_2 and we want to prove that $\forall y < |x| \alpha(x, y)$ is ∇_{i+1} in T^i_2. So α is defined by some machine e. We can construct without problems a machine f that on input x runs e for all inputs (x, y) with $y \leq |x|$ and checks of in all cases $\{e\}(x, y) = 0$. We know that in the standard model, but we have to prove it in T^i_2. In particular we need to show that a computation of f consists of $|x|$ runs of the machine e. So instead we first prove it for quasicomputations. Then take a computation of f. Like every quasicomputation, it does consist of $|x|$ quasicomputations of e on (x, y). But since the entire computation is a quasicomputation that follows the oracle, so are the pieces, hence they are also computations. Thus f does what it should.

Using the new notation we can interpret the proof of Corollary 4.28 as consisting of the following two parts.

Lemma 1 *If Φ is strictΔ^b_{i+1} in T^i_2, then it is also ∇_{i+1}.*

Hint: $\varphi(x,\{\bar{e}\}_{i+1}(x))$ is, clearly, ∇_{i+1} for $\varphi \in \Pi^b_i$.

Lemma 2 *T^i_2 proves induction for every ∇_{i+1} formula.*

We also need:

Lemma 3 (strengthening of Corollary 4.25) *Let T, M, K be as in the Corollary. Then K is ∇_{i+1} elementary substructure of M.*

Proof-sketch: Given a machine e with a Σ^b_i oracle we can define a machine e' which for every input x outputs the computation of e on x. Since K is $\Box^p_{i+1}$ closed, if x is in K, then the computation of $\{e\}_{i+1}(x)$ is also in K. Since quasicomputations are Σ^b_i definable and by Cor. 4.25, K is Σ^b_i elementary, the concept of a quasicomputation is absolute. But oracle queries are also Σ^b_i, hence the concept of computation is absolute too. $\Box$

For a formula α let $LOG-MIN(\alpha)$ be the following minimization principle:

$$\alpha(|z|,y) \rightarrow \exists x \leq |z| \ (\alpha(x,y) \wedge \forall x' < x \ \neg\alpha(x',y))$$

Note that $LOG-MIN$ is weaker than $LMIN$. ($LMIN\nabla_{i+1}$ entails $LMIN\Pi_i$ which axiomatizes S^{i+1}_2, hence it cannot be proved in T^i_2, unless Polynomial Hierarchy collapses.)

Lemma 4 *T^i_2 proves $LOG-MIN(\alpha)$ for every formula $\alpha(x,y)$ which is ∇_{i+1} in T^i_2.*

Proof-idea: Given a machine for α, construct a machine e which on input (z,y) constructs a sequence s of length $|z|+1$ such that $(s)_t = 0$ if $\neg\alpha(t,y)$ and $(s)_t = 1$ otherwise. The fact that machine e does what it should is again proved using quasicomputations. Once we have the sequence, we only need Δ_1 induction to find the least t such that $(s)_t = 1$. $\Box$

After we prove the key Theorem 4.32, we can eliminate ∇_{i+1} and the restriction *'strict'* because of the following two facts.

Lemma 5 *For every formula α which is Δ^b_{i+1} in T^i_2 there exists a formula α' which is strictΔ^b_{i+1} in T^i_2.*

Proof: Suppose $\alpha, \beta \in \Sigma^b_{i+1}$ and $T^i_2 \vdash \alpha \equiv \neg\beta$. By Cor. 4.12, there are strictΣ^b_{i+1} formulas α', β' such that $S^{i+1}_2 \vdash \alpha \equiv \alpha', \beta \equiv \beta'$. Consider the following implications.

$$\begin{array}{ccc} \neg\alpha & \rightarrow & \beta' \\ \uparrow & & \downarrow \\ \beta & \leftarrow & \neg\alpha' \end{array}$$

Note that $\uparrow$ is in T_2^i, the others are in S_2^{i+1}. We shall use $\forall\Sigma_{i+1}^b$ conservativity, Cor. 4.34, to prove that they are also provable in T_2^i. The $\rightarrow$ and $\leftarrow$ are of the form $\Pi_{i+1}^b \rightarrow \Sigma_{i+1}^b$ so they are Σ_{i+1}^b. The $\downarrow$ is $strict\Sigma_{i+1}^b \rightarrow strict\Pi_{i+1}^b$, so it can be written as $strict\Pi_{i+1}^b$. Because of the strictness this is $\forall\Sigma_i^b$. $\square$

Corollary 6 *T_2^i proves (1) induction for all Δ_{i+1}^b formulas, (2) $LOG-MIN$ for all Δ_{i+1}^b formulas.*

Note that some authors express (1) by "$T_2^i \vdash I\Delta_{i+1}^b$". However, in our book $I\Delta_{i+1}^b$ is a stronger schema, so I only know that $S_2^{i+1} \vdash I\Delta_{i+1}^b$, see Theorem 4.6. (The schema $I\Delta_{i+1}^b$ is $\forall\Sigma_{i+2}^b$, so we cannot use conservativity.)

Corollary 7 *Δ_{i+1}^b in T_2^i and ∇_{i+1} in T_2^i are the same classes.*

Proof: By Lemmas 1 and 5. $\square$

page 343, Proof of Theorem 4.29. To obtain formula (d.1) in T_2^i we have to use Lemma 3 above.

pages 345-347, Proof of Lemma 4.30. We have to use ∇_{i+1} instead of Δ_{i+1}^b throughout the proof. In formula (d.4) x should be corrected to c. The paragraph next after is completely wrong. We need to prove that (d.4) with parameter d included explicitly is ∇_{i+1}. To this end we use a universal machine that simulates all machines e for $e < |r|$ and checks $\psi(\{e\}_{i+1}(c))$. We can use overspill according to Lemma 4 above. To construct e_0 we argue in the same way. However, to prove that it is ∇_{i+1} we have to replace it by a standard machine and include the time bound into the input (instead of including the exponent it into the description of the machine).

page 457, line -21. Matiyaseviç should be Matiyasevič