4 Selected Solutions

- Exercise 7.25 a)

Solution. By definition we have $\mathsf{FIN} = \{x \mid W_x \text{ is finite}\}$ and

$$\mathsf{COF} = \{x \mid W_x \text{ is co-infinite}\} = \{x \mid {\sim}W_x \text{ is infinite}\} .$$

In order to show $\mathsf{FIN} \leqslant_{\mathsf{m}} \mathsf{COF}$ we define a total recursive function f such that $x \in \mathsf{FIN}$ if and only if $f(x) \in \mathsf{COF}$.

To describe the function f, we fix a recursive function φ_x and then define based on φ_x a recursive function φ_y such that $f(x) = y$. The construction will be such that we can read off the definition of f and verify that f is total recursive.

It clarifies the argument if we work with Turing machines instead of recursive functions directly. We assume that TM M computes φ_x and define TM N that computes φ_y. Thus it suffices to describe N in term of M such that the description of N is obtained from M in an effective way. We suppose M, N are defined over the alphabet $\Sigma = \{0, 1\}$ tacitly assuming the strings over Σ represent binary encodings of natural numbers.

We describe N: On input y, N generates all strings x, z such that $|x| > |y|$, $|z| > |y|$ and simulates M on x for at most $|z|$ steps. This is done in a timeshared manner such that eventually all computations $M(x)$ are simulated. N accepts its input y, if M ever accepts x (in at most $|z|$ steps). Then $\mathrm{L}(M)$ is finite implies that $\mathrm{L}(N)$ is finite, too. On the other hand if $\mathrm{L}(M)$ is infinite, then $\mathrm{L}(N) = \Sigma^*$. From this it is easy to see, that $x \in \mathsf{FIN}$ if and only $y \in \mathsf{COF}$.

It remains to verify that there exists a recursive function f such that $f(x) = y$. To see this, observe that x is nothing else then the code of M, where N calls M as a subroutine. Then the code of N is essentially given as the code of the above loop plus the code of M, x. It is easy to see that this process can be represented by a recursive function. □

- Exercise 7.25 b)

Solution. By definition we have $\mathsf{INF} = \{x \mid W_x \text{ is infinite}\}$ and we have to show $\mathsf{INF} \leqslant_{\mathsf{m}} \mathsf{COF}$. For that we employ computation histories. Recall that a string represents a computation history of a given TM M if it satisfies the following properties:

1. The string encodes a sequence of configurations of M.
2. The first configuration is the start configuration on some input.
3. The last configuration is an accepting configuration.
4. The $i + 1^{\text{th}}$ configuration follows from the i^{th} configuration in accordance with the rules of M.

Let M be a TM, then we aim to construct a TM N such that if $\mathrm{L}(M)$ is infinite, then ${\sim}\mathrm{L}(N)$ will be infinite and otherwise if $\mathrm{L}(M)$ is finite, then ${\sim}\mathrm{L}(N)$ will be finite. If this has been achieved then we can define a recursive function f from this construction (as in Exercise 7.25.a) such that $x \in \mathsf{INF}$ if and only if $f(x) \in \mathsf{COF}$.

We define N to accept all strings that are not computation histories of M. The simplest way to do so is to extend the alphabet Σ such that the encoding can be done with the new symbols. Then we only need to check whether one of the above given properties fails to hold. In which case N will accept, otherwise reject. □

- Exercise 7.25 c)

Solution. Let $\mathsf{TOT} = \{x \mid \varphi_x \text{ is total}\}$, we have to show $\mathsf{TOT} \leqslant_{\mathsf{m}} \mathsf{COF}$. The reduction makes use of an auxiliary TM K, defined as follows.

On input y. K generates all strings x (over Σ) such that $|x| \leqslant |y|$. Then K simulates M on x (in a timeshared way so that all x generated are eventually tested). If M halts on all x, then K accepts y. Notice, that if $\mathrm{L}(M)$ is total, then M halts for all x. Hence for all input y to K, K will accept its input. Hence $\mathrm{L}(K) = \Sigma^*$. Otherwise, if there exists x such that M loops on x. then K will reject all y with $|y| \geqslant |x|$. In particular $\mathrm{L}(K)$ is finite.

Now, to define the sought TM N, we construct a TM that accepts all strings that are not configuration histories of K (employing ideas from Exercise 7.25 b). I.e., $\sim\mathrm{L}(N)$ is infinite if $\mathrm{L}(M)$ is total and $\sim\mathrm{L}(N)$ is finite if $\mathrm{L}(M)$ is not total. It is easy to see how to prove $\mathsf{TOT} \leqslant_{\mathsf{m}} \mathsf{COF}$ from this. □

- Exercise 7.25 d)

Solution. Let $\mathsf{REC} = \{x \mid W_x \text{ is recursive}\}$. According to the exercise we ought to prove $\mathsf{COF} \leqslant_{\mathsf{m}} \mathsf{REC}$. However this would contradict the fact that the arithmetical hierarchy doesn't collapse.

To see this, observe that COF can be represented as follows:

$$\begin{aligned} \mathsf{COF} &= \{M \mid \mathrm{L}(M) \text{ is co-infinite}\} \\ &= \{M \mid \forall n \exists x \forall t\ (|x| > n \land\ M \text{ doesn't accept } x \text{ in } t \text{ steps})\}\ . \end{aligned}$$

Clearly the assertion that "M doesn't accept x in t steps" is representable as recursive formula. We can even prove that this formula is primitive recursive. Hence the set COF represents a Π_3-formula. Moreover it is not difficult to proof that COF is complete for Π_3 with respect to $\leqslant_{\mathsf{m}}$. On the other hand we have the following characterisation of REC:

$$\begin{aligned} \mathsf{REC} &= \{M \mid \mathrm{L}(M) \text{ is recursive}\} \\ &= \{M \mid \exists N \forall x \forall t_1 \exists t_1 \exists t_2\ (M \text{ accepts } x \text{ in } t_1 \text{ steps} \rightarrow N \text{ accepts } x \text{ in } t_2 \text{ steps})\}\ . \end{aligned}$$

From this it is not difficult to see that $\mathsf{REC} \in \Sigma_3$. If we would indeed by able to reduce prove $\mathsf{COF} \leqslant_{\mathsf{m}} \mathsf{REC}$, then we would reduce *all* Π_3-formula to a Σ_3-formulas. Hence the arithmetical hierarchy collapses at level 3 which is not the case. □