**Exercises for Computability and Complexity, Spring 2019, Sheet 9 – Solutions**

*Please return your solutions in the Tuesday lecture on April 23*

This exercise sheet revolves around the problem PARTITION. This is a standard NP problem which is defined as follows:

The instances of PARTITION consist in a finite nonempty set $A$ and a function $s: A \to \mathbb{N}$ (the intuition is that $s$ assigns to each element $a \in A$ a "size" $s(a)$). The question is: can $A$ be split into two disjoint subsets $A = B \cup C$, such that the total sizes of $B$ and $C$ are the same, that is $\sum_{b \in B} s(b) = \sum_{c \in C} s(c)$?

**Exercise 1** Describe a coding scheme by which instances of PARTITION can be represented as input words to a TM, using the coding alphabet {0, 1, #}.

**Solution.** One among an infinite choice of coding options is to code an instance of PARTITION, where $A = \{a_1, ..., a_k\}$, by a the list of integers $s(a_1), ..., s(a_k)$, where each $s(a_i)$ is rendered by the 0-1-string writing $s(a_i)$ to base 2, and where these integer strings are separated by #.

**Exercise 2** Show that PARTITION is in NP (use the input coding that you proposed in Exercise 1). Be reasonably precise in describing your solution, to the level that you describe what intermediate results a TM will write on which of its tapes.

**Solution.** A nondeterministic TM $M$ deciding PARTITION using the above coding can be designed as follows. $M$ operates in a series of stages. $M$ has 4 tapes besides the input tape (numbered 1, 2, 3, 4, 5; 1 is the input tape).

Stage 1: $M$ sweeps once over the input from left to right. Whenever it starts reading a new integer, it copies it either on tape 2 or on tape 3 (a random decision); separating copied integer strings by # on these two tapes. Time needed: $n$ +1 (where $n$ is the length of the input).

Stage 2: $M$ computes the sum of the integers on tape 2, writing this sum on tape 4. This can be done in time O([length-of-inscription-on-tape2]$^2$) = O($n^2$). Then $M$ does the same for the integers found on tape 3, writing their sum on tape 5. Total time for stage 2: O($n^2$).

Stage 3: $M$ compares the binary words now found on tapes 4 and 5. If they are identical, accept, else reject. Time needed: O($n$).

It is clear that an accepting run of this kind exists if and only if the correct decision is "yes".

The total time for all three stages is O($n^2$), hence PARTITION is in NP.