Algorithms and Complexity. Exercise session 7

Undecidability

Undecidability 1 Is there an explicit program P so that for a given y it is decidable whether P terminates on input y?

Undecidability 2 Is there an explicit program P so that for a given y it is undecidable whether P terminates on input y?

Undecidability 3 If the program x terminates on empty input, we denote by $f(x)$ the number of steps before terminating. Otherwise, we set $f(x) = 0$. Define now $MR(y)$, the maximum running time of all programs whose binary encoding is less than y.

$$MR(y) = \max_{x \le y} f(x),$$

Is MR computable?

Undecidability 4 Show that the function MR in the previous exercise grows faster than any recursive function. In particular, show that for every recursive function g there is a y such that $MR(y) > g(y)$.

Undecidability 5 Suppose you are given a Turing machine M, an input X (which is on the tape from the beginning) and an integer constant K. Is the following problem decidable or not?

Does M terminate on input X using at most K cells of the tape (a used cell may written and read several times)?

Construct knapsack Knapsack problem is a well-known NP-complete problem. The input is a set P of objects with weight w_i and the value v_i, the knapsack size S and a target K. The question is to determine if it is possible to pick a subset of objects of value less than K, such that their total weight does not exceed S. All numbers in input are non negative integers.

Now assume that we have an algorithm A that solves the above decision problem. Construct an algorithm that uses A to solve the constructive knapsack problem, ie. given the same input as A it answers the knapsack problem and secondly, it tells you exactly which objects to pack into the knapsack. The algorithm may call A $O(|P|)$ times, so it can not take more than polynomial time.

Then construct and analyze a Turing reduction of the constructive knapsack problem to the usual knapsack problem (which is a decision problem).
