# The Theory of Computation

Bernard M.E. Moret

Addison-Wesley-Longman, 1998

## *Solution Manual*
## *Chapter 6, Problem 6.15*

That constant functions are both time- and space-constructible is obvious: for each fixed constant $c$, it is easy to construct a Turing machine that ignores its input and makes exactly $c$ steps (or writes exactly $c$ tape squares) before halting.

For polynomial and exponential functions, we assume that all coefficients are positive, since we cannot in any case represent negative values.

That polynomial and exponential functions are space-constructible is almost as easy: we can easily construct a Turing machine that uses exactly as much space as its input (by copying its input tape onto its scratch tape), any fixed multiple of its input (by encoding in the finite state the number of copies), the square of its input (by keeping on the scratch tape a counter into the input tape, effectively marking each square for each copy made—the counter itself is no longer than the first copy and can be kept in the space used by the first copy), any power of its input (repeat the squaring or multiplication), any integer constant larger than 1 raised to the power of its input (similar tricks), and thus any polynomial or exponential function. For time construction, we can certainly do any simple linear functions (just spend as many steps per input symbol as needed) and any first-degree polynomial (just add the constant extra number of steps through the finite-state control). For higher powers and exponentials, we need to count, which will by itself consume some steps; for instance, the simplest way to construct a machine for large-valued functions is to evaluate the function at the given argument, set up a counter with that value, and start decrementing the counter. For each specific function we can easily compute in advance the amount of overhead incurred in the various maintenance operations and thus compute an appropriately decreased function.