Universal Turing Machine

A Turing machine that is able to simulate any other Turing machine is called a *Universal Turing machine.*

The concept of the Turing machine is based on the idea of a person executing a well-defined procedure by changing the contents of an unlimited paper tape, which is divided into squares that can contain one of a finite set of symbols. The person needs to remember one of a finite set of states and the procedure is formulated in very basic steps in the form of "*If your state is 42 and the symbol you see is a '0' then replace this with a '1', move one symbol to the right, and assume state 17 as your new state.*

A TAPE which is divided into cells, one next to the other. Each cell contains a symbol from some finite alphabet. The alphabet contains a **special blank symbol** (here written as 'B') and one or more other symbols. The tape is assumed to be arbitrarily extendable to the left and to the right, i.e., the Turing machine is always supplied with as much tape as it needs for its computation. Cells that have not been written to before are assumed to be filled with the blank symbol. In some models the tape has a left end marked with a special symbol; the tape extends or is indefinitely extensible to the right. The symbols are sometimes referred to as colors.

A HEAD that can read and write symbols on the tape and move the tape left and right one (and only one) cell at a time. In some models the head moves and the tape is stationary.

A TABLE ("action table", or transition function) of instructions (usually quintuples or 5-tuples but sometimes 4-tuples) that, given the state the machine is currently in and the symbol it is reading on the tape tells the machine to do the following in sequence (for the 5-tuple models): (i) **either erase or write** a symbol, and then (ii) move the head ('L' for one step left or 'R' for one step right), and then (iii) assume the same or a new state as prescribed. In the 4-tuple models the TABLE tells the machine to (ia) erase or to write a symbol or (ib) move the head left or right, and then (ii) assume the same or a new state as prescribed, but not both actions (ia) and (ib) in the same instruction. In some models, if there is no entry in the table for the current combination of symbol and state then the machine will halt; other models require all entries to be filled.

Turing machines are extremely basic abstract symbol-manipulating devices which, despite their simplicity, can be adapted to simulate the logic of any computer that could possibly be constructed. They were described in 1936 by Alan Turing. Though they were intended to be technically feasible, **Turing machines were *not* meant to be** a practical computing technology, but a thought experiment about the limits of mechanical computation; thus they were not actually constructed. Studying their abstract properties yields many insights into computer science and complexity theory.

A state register that stores the state of the Turing table. The number of different states is always finite and there is one special start state with which the state register is initialized. Turing defined this as a "note of instructions" to preserve the computation of the "computer" (a person) who is working in a "desultory manner":

DISPLAY REGULAR

Steam punk ABCD hgovs

DISPLAY BOLD

Steam punk ABCD hgovs

DISPLAY BLACK

Steam punk ABCD hgovs

DISPLAY ITALIC

Steam punk ABCD hgovs

DISPLAY BOLD ITALIC

Steam punk ABCD hgovs

DISPLAY BLACK ITALIC

Steam punk ABCD hgovs

TEXT REGULAR

Steam punk ABCD hgovs

TEXT BOLD

Steam punk ABCD hgovs

TEXT ITALIC

Steam punk ABCD hgovs

TEXT BOLD ITALIC

Steam punk ABCD hgovs

CAPTION REGULAR

Steam punk ABCD hgovs

CAPTION BOLD

Steam punk ABCD hgovs

CAPTION ITALIC

Steam punk ABCD hgovs

CAPTION BOLD ITALIC

Steam punk ABCD hgovs