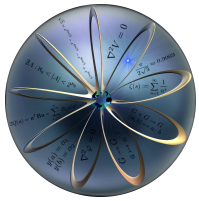

# planetmath.org

Math for the people, by the people.

## Turing machine

| | |
|---|---|
| Canonical name | TuringMachine |
| Date of creation | 2013-03-22 12:59:01 |
| Last modified on | 2013-03-22 12:59:01 |
| Owner | Henry (455) |
| Last modified by | Henry (455) |
| Numerical id | 7 |
| Author | Henry (455) |
| Entry type | Topic |
| Classification | msc 68Q10 |
| Classification | msc 68Q05 |
| Classification | msc 03D10 |
| Related topic | DeterministicTuringMachine |
| Related topic | NonDeterministicTuringMachine |

A *Turing machine* is an imaginary computing machine invented by Alan Turing to describe what it means to compute something.

The "physical description" of a Turing machine is a box with a tape and a tape head. The tape consists of an infinite number of cells stretching in both directions, with the tape head always located over exactly one of these cells. Each cell has one of a finite number of symbols written on it.

The machine has a finite set of states, and with every move the machine can change states, change the symbol written on the current cell, and move one space left or right. The machine has a program which specifies each move based on the current state and the symbol under the current cell. The machine stops when it reaches a combination of state and symbol for which no move is defined. One state is the start state, which the machine is in at the beginning of a computation.

A Turing machine may be viewed as computing either a partial function or a relation. When viewed as a function, the tape begins with a set of symbols which are the input, and when the machine halts, whatever is on the tape is the output. For instance it is not difficult to write a program which doubles a binary number, so input of 10 (with 0 on the first cell, 1 on the second, and all the rest blank) would give output 100. If the machine does not halt on a particular input then the function is undefined on that input.

Alternatively, a Turing machine may be viewed as computing a relation. In that case the initial symbols on the tape is again an input, and some states are denoted "accepting." If the machine halts in an accepting state, the symbol is accepted, if it halts in any other state, the symbol is rejected. A slight variation is when all states are accepting, and a symbol is rejected if the machine never halts (of course, if the only method of determining if the machine will halt is watching it then you can never be sure that it won't stop at some point in the future).

Another way for a Turing machine to compute a relation is to list (enumerate) its members one by one. A relation is recursively enumerable if there is some Turing machine which can list it in this way, or equivalently if there is a machine which halts in an accepting state only on the members of the relation. A relation is recursive if it is recursively enumerable and its complement is also. An equivalent definition is that there is a Turing machine which halts in an accepting state only on members of the relation and always halts.

There are many variations on the definition of a Turing machine. The

tape could be infinite in only one direction, having a first cell but no last cell. Even stricter, a tape could move in only one direction. It could be two (or more) dimensional. There could be multiple tapes, and some of them could be read only. The cells could have multiple tracks, so that they hold multiple symbols simultaneously.

The programs mentioned above define only one move for each possible state and symbol combination; these are called deterministic. Some programs define multiple moves for some combinations.

If the machine halts whenever there is any series of legal moves which leads to a situation without moves, the machine is called non-deterministic. The notion is that the machine guesses which move to use whenever there are multiple choices, and always guesses right.

Yet other machines are probabilistic; when given the choice between different moves they select one at random.

No matter which of these variations is used, the recursive and recursively enumerable relations and functions are unchanged (with two exception–one of the tapes has to move in two directions, although it need not be infinite in both directions, and there can only be a finite number of symbols, states, and tapes): the simplest imagineable machine, with a single tape, one-way infinite tape and only two symbols, is equivalent to the most elaborate imagineable array of multidimensional tapes, lucky guesses, and fancy symbols.

However not all these machines can compute at the same speed; the speed-up theorem states that the number of moves it takes a machine to halt can be divided by an arbitrary constant (the basic method involves increasing the number of symbols so that each cell encodes several cells from the original machine; each move of the new machine emulates several moves from the old one).

In particular, the question $\mathcal{P} \stackrel{?}{=} \mathcal{NP}$, which asks whether an important class deterministic machines (those which have a polynomial function of the input length bounding the time it takes them to halt) is the same as the corresponding class of non-deterministic machines, is one of the major unsolved problems in modern mathematics.