Turing Machines

The focus of the course throughout the entire semester has been the design and analysis of efficient computer algorithms. So far, we have been able to find efficient algorithms to virtually all of the problems we have examined; however, there is a class of problems that we consider here for which no know efficient computer algorithm exists. Does this mean it is not possible to find such an algorithm? As we will see, the answer to that question still eludes us.

Prior to examining this class of problems, we need some background theoretical material to help in the analysis of the problems. We start this discussion by considering a fundamental model of computation—the Turing machine.

Deterministic Turing Machines

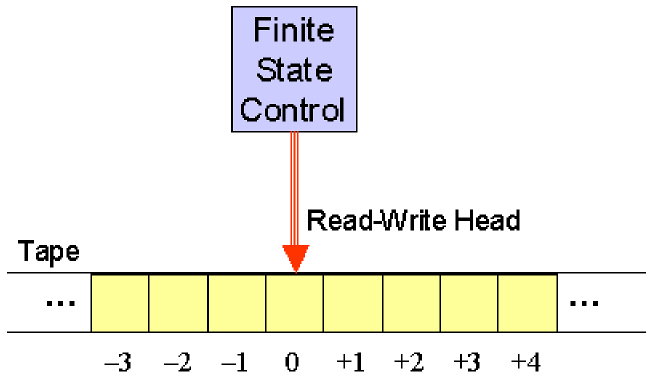
Definition: A deterministic Turing machine (DTM) consists of the following:

- A finite state control,
- A tape consisting of a two-way sequence of tape squares, and
- A read-write head.

A given DTM is manipulated through the execution of a program utilizing the following components:

- A finite set Γ of symbols,
- A finite set Q of states (q_0 is designated as the start state and q_Y and q_N are designated as halting states), and
- A transition function $\delta : (Q - \{q_Y, q_N\}) \times \Gamma \to Q \times \Gamma \times \{+1, -1\}$.

In other words, the DTM will scan the tape, and the transition function will read the symbol from Γ currently written on the table and determine what character to write in its place as well as which direction to move the read-write head. Presumably, if the program indicates $+1$, the head moves to the right. On the other hand, if the program indicates -1, the head moves to the left. A DTM is depicted graphically as follows:



To see how a DTM works, suppose we have $\Sigma \subset \Gamma$ as a set of input symbols and $b \in \Gamma - \Sigma$ correspond to the "blank" symbol. Next suppose we have an input string $x \in \Sigma^*$ where we place the symbols of x in consecutive cells of the tape in positions $1 \ldots |x|$. All other cells on the tape are assumed to have b. The program will begin in state q_0 with the read-write head scanning square 1 and proceed according to the transition function.

If the current state of the DTM is ever q_Y or q_N, then the program terminates. On the other hand, if the current state is in $Q - \{q_Y, q_N\}$, then the program continues. When transitioning, the read-write head will first erase the symbol in the square it is examining and then write the new symbol as specified by the transition function. The read-write head then either moves one position to the right or one position to the left, and the Finite State Control updates the state to some successor q'.

Because we have limited the set of halting (or terminal) states to $\{q_Y, q_N\}$, we note that a DTM is only able to solve problems that return a Boolean result. In particular, we say that a DTM is used to solve "decision problems" where q_Y indicates the DTM has decided "yes" and q_N indicates the DTM has decided "no."

Computability

The Turing machine provides the foundation for determining what is computable. In fact, Turing's thesis is that any computation that can be carried out by mechanical means can also be carried out on a Turing machine. This is a rather remarkable claim in that virtually all of computer science can be summed up in the capabilities of a rather simple model.

To formalize this notion, we will use the symbol "⇒" to denote a computation (or a transition from an input state to some output state) within a Turing machine.

Definition: A function f with domain D is Turing computable (or simply computable) if there exists a Turing machine M such that

$$q_o x \Rightarrow^* q_f f(x).$$

In other words, f is computable if there exists a Turing machine M such that M halts on input x.