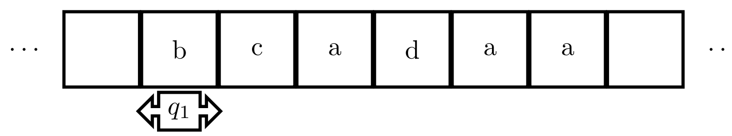
Turing Machines

Kevin Li

April 2023

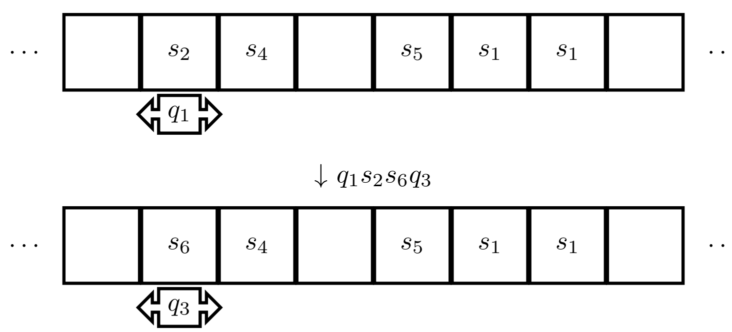
1 Introduction

Today we are going to study one of the most commonly used models of computation: the Turing machine. This was invented by Alan Turing and looks something like the following:

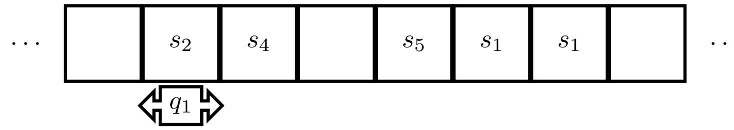


The row of squares is known as a **tape**. It is infinite in both directions, and each square is either blank or contains a symbol from a fixed finite set $\{s_1, \ldots, s_n\}$. We will denote a blank square with B. Below this tape is a moving pointer called the **reading head**. At any given time, it is *reading* the square above it, and is always in some state, q_i. There are finitely many states $\{q_1, \ldots, q_m\}$.

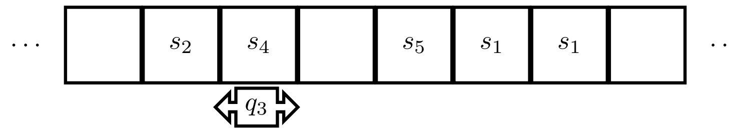
The machine is capable of three instructions: 1. If the machine is in state q_i and reading s_j, then the machine can erase the symbol s_j, write the symbol s_k, and change its state to q_l. This instruction is written as $q_i s_j s_k q_l$.



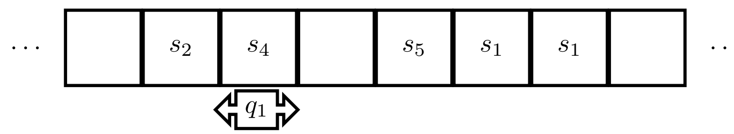
2. If the machine is in state q_i reading s_j, then the head can move one square right and change state to q_k. This instruction is written as $q_i s_j R q_k$.



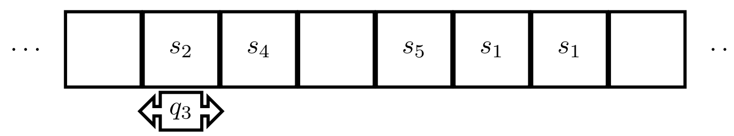
$\downarrow q_1 s_2 R q_3$



3. If the machine is in state q_i reading s_j, then the head can move one square left and change state to q_k. This instruction is written as $q_i s_j L q_k$.



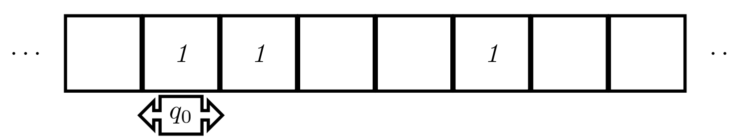
$\downarrow q_1 s_4 L q_3$



Definition 1 *A **Turing Machine Program** for machine M is a finite list of instructions such that for any pair $q_i s_j$, there is at most one instruction of the form $q_i s_j \alpha q_k$. We will denote this program as P.*

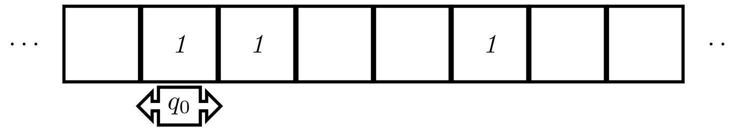
Definition 2 *An **Initial State** for Turing machine program P is the starting tape for which the program P will run. Note this includes what symbols are on the tape, the location of the reading head, and the state it is in.*

Example 1 *Consider a Turing machine M with alphabet $\{1\}$ (remember, all alphabets 'contain' the blank symbol B). Let P be the program:*
$q_0 1 L q_0$
$q_0 B 1 q_1$
$q_1 1 R q_2$.
Consider the initial state:

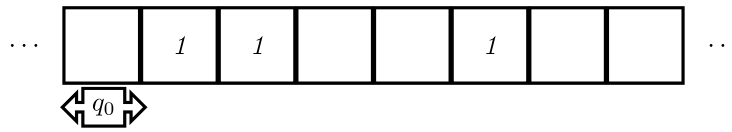


Then, the program will run as follows:

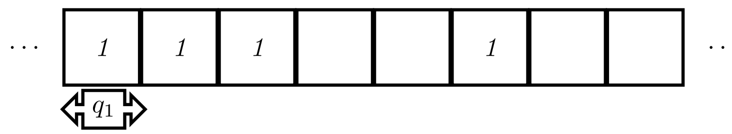
Initial state:



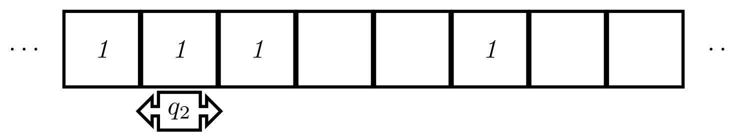
Next: $q_0 1 L q_0$



Next: $q_0 B 1 q_1$



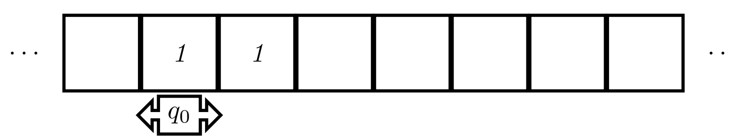
Next: $q_1 1 R q_2$



Notice that there are no instructions of the form $q_2 1 \beta \gamma$.

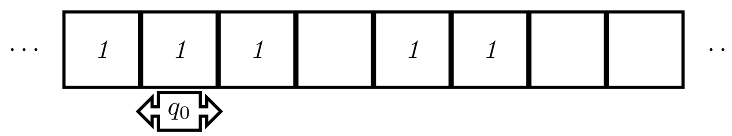
Definition 3 *If the Turing machine is running program* P, *instate* q_i *and reading symbol* s_j, *if there is no instruction of the form* $q_i s_j \alpha q_l$, *then we say the machine is in* ***halting state***.

Problem 1 *Consider a Turing machine with alphabet* $\{1\}$ *and states* $\{q_0, q_1, q_2\}$. *Consider the following program:*
$q_0 1 R q_0$
$q_0 B 1 q_1$
$q_1 1 R q_2$
Show all steps of the computation of this program on initial state:



Problem 2 *Is there a program for a Turing machine with alphabet* $\{1\}$ *which never reaches halting state on some initial state? You can choose what states you have.*

Problem 3 *Construct a Turing machine program with the alphabet* $\{1\}$ *(using any finite states you want including* q_0*) which writes* 1 *onto the first two blanks to the right of the initial reading head in intial state:*



Does this program reach halting state for every tape that has at least 2 blanks to the right of the initial reading head? If yes, try to explain why. If not, try to write a program which does.

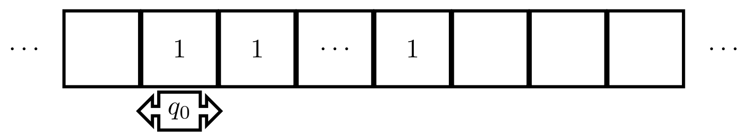
2 Turing Computable Function

We want to think of Turing machines as a computation method, similar to a computer. Let's try to formalize how Turing machines relate to certain functions in mathematics. For the purposes of this worksheet, $0 \in \mathbb{N}$.

Definition 4 *A partial function $f : X \to Y$ is a function whose domain is a subset $D \subseteq X$ and its codomain is Y. In other words, a partial function f is a function which only produces an output on some (possibly all) input elements. For any $x \in X$, if $x \in D$ then we say $f(x)$ converges, written $f(x)\downarrow$. Otherwise, we say $f(x)$ diverges, written $f(x)\uparrow$.*

Example 2 *$f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = \sqrt{x}$ is a partial function, since is domain is $\{x \in \mathbb{R} : x \geq 0\}$.*

Let 1 be in our alphabet. Let P be a Turing machine program. The partial function $f : \mathbb{N} \to \mathbb{N}$ is computed as follows:
To compute $f(x)$, start with initial state



with $x + 1$ many $1's$. Then, run program P.
If this program halts, we say $f(x) \downarrow=$ the number of $1's$ in the halting state
If this program does not halt, we say $f(x)\uparrow$.

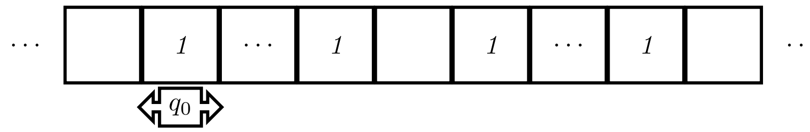
Definition 5 *We say a partial function $f : \mathbb{N} \to \mathbb{N}$ is **Turing computable** if there is a Turing machine and a program which computes it.*

Problem 4 *Show that the function $f(x) = x + 2$ is Turing computable.*

Problem 5 *Show that the function $f(x) = x + n$ is Turing computable for each $n \in \mathbb{N}$.*

Problem 6 *Show that the function $f(x) = n$ is Turing computable for each $n \in \mathbb{N}$ (i.e. show that constant functions are Turing computable).*

Definition 6 *We say a function $f : \mathbb{N}^2 \to \mathbb{N}$ is Turing computable if on input $(x, y) \in \mathbb{N}^2$, there is a Turing machine program which computes $f(x, y)$ on the initial state*



where the first string has $x + 1$ many $1'$s and the second string has $y + 1$ many $1'$s.

Problem 7 *Show that the function $f(x, y) = x + y$ is Turing computable.*

Problem 8 *Show that the functions $\pi_1^2(x, y) = x$ and $\pi_2^2(x, y) = y$ are Turing computable.*

Problem 9 *Revisit the program in Problem 1. What function does this program compute?*

Problem 10 *Choose one of the following: 1. Think of any function which you can calculate in your head. Try to write a Turing machine program which computes it. 2. Ask your instructors about 'The Halting Problem'.*