

Introduction

- Computer science = study of algorithms
- In this course we will study the limits of algorithms:
 - Are there problems which cannot be solved by any algorithm?
 - How quickly can a problem be solved?
- For this, we need a formal definition of “algorithm”

Turing machines

- Invented by Alan Turing in 1936
- The physical device consists of:
 - A finite state control
 - A tape divided into squares
 - A moveable tape-head that is positioned over a square of the tape.
- A Turing machine is specified by a tuple $(Q, \Gamma, \Sigma, q_0, q_{accept}, q_{reject}, \delta)$ consisting of
 - A finite set Q of states
 - An tape alphabet Γ containing a special symbol \sqcup (called “blank”)
 - An input alphabet $\Sigma \subseteq \Gamma \setminus \{\sqcup\}$
 - An initial state $q_0 \in Q$
 - Accept and reject states $q_{accept}, q_{reject} \in Q$
 - A transition function $\delta : Q \setminus \{q_{accept}, q_{reject}\} \times \Gamma \mapsto Q \times \Gamma \times \{L, R\}$
- A Turing machine receives an input string $w \in \Sigma^*$ as follows:
 - w is written on the tape; all other squares contain \sqcup
 - The tape-head is positioned over the left-most symbol of w
 - The Turing machine begins in state q_0
- Thereafter the Turing machine computes one step at a time. In each step:
 - Let q be the current state of the machine
 - Let a be the symbol under the tape-head
 - Let $(q', a', D) = \delta(q, a)$
 - The Turing machine replaces a with a' , enters state q' , and moves the tape-head one square in the direction D (L = “left”, R = “right”)
- If the Turing machine is in state q_{accept} or q_{reject} , it halts

Example

- Construct a Turing machine that accepts a string over $\{0, 1\}^*$ iff it contains an equal number of 0's and 1's, and rejects otherwise.
- Idea: the tape alphabet is $\{0, 1, \times, \sqcup\}$. Move through the input from left to right and change the first 0 to a \times and the first 1 to a \times . If a 0 is found but not a 1, or a 1 is found but not a 0, reject. If neither a 0 nor a 1 is found, accept. Otherwise, repeat.
- $Q = \{q_0, q_1, q_2, q_3, q_{accept}, q_{reject}\}$
 - q_0 : “neither a 0 nor a 1 has been found”
 - q_1 : “a 1 has been found, but not a 0”
 - q_2 : “a 0 has been found, but not a 1”
 - q_3 : “both a 0 and a 1 have been found (rewind)”
- $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup, \times\}$
- δ is defined by the following table (note: I have written a * for the new symbol and the direction in case the new state is q_{accept} or q_{reject} . This is because we don't care about the actual values, since the TM stops computing once it reaches q_{accept} or q_{reject} . In order to be completely formal, though, we would need to specify some arbitrary value for these entries).

| STATE | SYMBOL | STATE | SYMBOL | DIRECTION |
|-------|----------|--------------|----------|-----------|
| q_0 | 0 | q_2 | \times | R |
| q_0 | 1 | q_1 | \times | R |
| q_0 | \times | q_0 | \times | R |
| q_0 | \sqcup | q_{accept} | * | * |
| q_1 | 0 | q_3 | \times | L |
| q_1 | 1 | q_1 | 1 | R |
| q_1 | \times | q_1 | \times | R |
| q_1 | \sqcup | q_{reject} | * | * |
| q_2 | 0 | q_2 | 0 | R |
| q_2 | 1 | q_3 | \times | L |
| q_2 | \times | q_2 | \times | R |
| q_2 | \sqcup | q_{reject} | * | * |
| q_3 | 0 | q_3 | 0 | L |
| q_3 | 1 | q_3 | 1 | L |
| q_3 | \times | q_3 | \times | L |
| q_3 | \sqcup | q_0 | \sqcup | R |