Turing Machines

Lecture 25
Section 9.1

Robb T. Koether

Hampden-Sydney College

Wed, Oct 26, 2016

Outline

Outline

Computation

- Can a DFA or a PDA “compute” that $1 + 1 = 2$?

Computation

- The nearest they can come is to read input of the form $a + b = c$, with a, b, and c in binary, and accept or reject it.

- The nearest they can come is to read input of the form $a + b = c$, with a, b, and c in binary, and accept or reject it.
- Accept the input $1 + 1 = 2$.

Computation

- The nearest they can come is to read input of the form $a + b = c$, with a, b, and c in binary, and accept or reject it.
- Accept the input $1 + 1 = 2$.
- Reject the input $1 + 1 = 3$.

Computation

- But this requires that we input the answer and that the machine simply confirms it.
- We want a machine that will read a and b and *produce* the sum $a + b$.

Outline

- Turing machines are far more powerful than DFAs or PDAs.
- A Turing machine is as powerful as any computer ever built or ever will be built (provided we are not concerned with efficiency).

Abilities of a Turing Machine

- A Turing machine is similar to a DFA or a PDA, with the following differences.
  - It can read and write to the tape.
  - It can move right or left on the tape.
  - It halts as soon as it reaches a state for which the next move is not defined for the current state and tape symbol.

The Transition Function

- The transition function will
  - Read the current tape symbol (the symbol at the current position) and then
  - Write a symbol to that position and
  - Move one position left or right.

The Transition Function

- The transition function will
  - Read the current tape symbol (the symbol at the current position) and then
  - Write a symbol to that position and
  - Move one position left or right.
- The inputs are
  - The current state
  - The current tape symbol

The Transition Function

- The transition function will
  - Read the current tape symbol (the symbol at the current position) and then
  - Write a symbol to that position and
  - Move one position left or right.
- The inputs are
  - The current state
  - The current tape symbol
- The outputs are
  - The new state
  - The new tape symbol
  - Whether to move left or right

Turing Machines

- A Turing machine may be viewed as a computer program.
- The input is the initial contents of the tape.
- The output is the final contents of the tape.

Looping

- Because we can move right or left on the input tape, it is possible that the machine will never halt.
- If this happens, we will say that the Turing machine loops.
- How do we detect looping?

Looping

- Because we can move right or left on the input tape, it is possible that the machine will never halt.
- If this happens, we will say that the Turing machine loops.
- How do we detect looping?
- Good question!

Looping

- Because we can move right or left on the input tape, it is possible that the machine will never halt.
- If this happens, we will say that the Turing machine loops.
- How do we detect looping?
- Good question!
- That is the Halting Problem.

Definition of a Turing Machine

Definition (Turing machine)

A Turing machine is a 7-tuple $(Q, \Sigma, \Gamma, \delta, \square, q_0, F)$, where

- Q is a finite set of states,
- Σ is the finite input alphabet,
- Γ is the finite tape alphabet,
- δ is the transition function,
- $\square \in \Gamma$ is the blank,
- $q_0 \in Q$ is the start state,
- $F \subseteq Q$ is the set of final states.

We require that $\Sigma \subseteq \Gamma$ and that $\square \notin \Sigma$.

The Transition Function

Definition (The Transition Function)

The transition function

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}.$$

Furthermore, δ is a partial function, meaning that it is defined only on a subset of $Q \times \Gamma$.

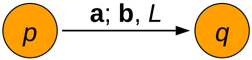
We leave δ undefined for some elements of $Q \times \Gamma$ so that the Turing machine will be able to halt.

The Tape

- The tape is infinite in both directions.
- The tape is initially filled with blanks.
- Then we write the (finite) input on the tape and begin processing.
- Processing begins in the start state with the read head positioned at a specified position which by default is the left end of the input.

Transitions

- Each transition
  - Begins in a state p,
  - Reads a symbol **a**,
  - Changes to a state q (possibly $q = p$),
  - Writes a symbol **b** (possibly **b** $=$ **a**), and
  - Moves left or right (L or R).
- We will represent the transition $\delta(p, \mathbf{a}) = (q, \mathbf{b}, L)$ as



The Input Alphabet

- Because we are now interested in Turing machines as computers, we will typically let our input alphabet be

$$\Sigma = \{\mathbf{0}, \mathbf{1}\}.$$

- That is, the input will be encoded in binary.

Outline

Example

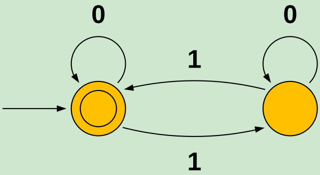
Example

Design a Turning Machine that will accept the language $L(\mathbf{0}^*(\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*)^*)$, that is, the language of all binary strings that contain an even number of **1**'s.

Example

Example

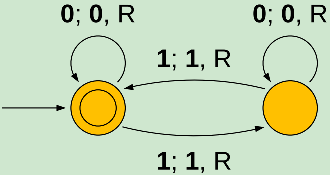
A DFA for this language is



Example

Example

A Turing machine that accepts this language is



Outline

Views of Turing Machines

- We may view Turing machines in a variety of ways.
  - Language accepter.
  - Language processor.
  - Language enumerator.
  - Function evaluator.
  - Problem decider.

Views of Turing Machines

Definition (Language accepter)

A language accepter is a Turing machine that reads an input w from the tape and, if w is accepted, moves to the accept state, and if w is rejected, either moves to the reject state or loops.

- From the previous example, it is easy to see how a Turing machine could be built that would accept a given regular language.
- How would we build a Turing machine to accept a context-free language?

Example ($L = \{\mathbf{a}^n\mathbf{b}^n \mid n \geq 0\}$)

Design a Turing machine that will accept the language

$$L = \{\mathbf{a}^n\mathbf{b}^n \mid n \geq 0\}.$$

Views of Turing Machines

Definition (Language processor)

A language processor is a Turing machine that begins with a word from a language on the tape and halts with a word from another language on the tape.

Example (Language processor)

The Turing machine COPY replaces the input w with the output $w\square w$.

Views of Turing Machines

Definition (Language enumerator)

A language enumerator is a Turing machine that begins with a blank tape and writes sequentially all the words in a language L.

Example (Language enumerator)

The Turing machine ENUM writes all the strings in $\{\mathbf{0}, \mathbf{1}\}^*$:

$\square\square\mathbf{0}\square\mathbf{1}\square\mathbf{00}\square\mathbf{01}\square\mathbf{10}\square\mathbf{11}\square\mathbf{000}\ldots$.

Views of Turing Machines

Definition (Function evaluator)

A function evaluator is a Turing machine that reads an input w from its tape and writes $f(w)$, for some function f.

Example (Function evaluator)

The Turing machine INCR replaces the input n with $n + 1$, thereby computing the function $f(n) = n + 1$.

Views of Turing Machines

Definition (Problem decider)

A problem decider is a Turing machine that reads a decision problem coded in binary and accepts the input if the answer is “yes” and rejects the input if the answer is “no.”

Example (Problem decider)

The Turing machine PRIME writes **1** if the input *n* is a prime number and writes **0** if it is not prime.

Outline

Assignment

Assignment

- Section 9.1 Exercises 2, 3, 5, 8fgh, 10, 13a