Week 6 at a glance

Textbook reading: Chapter 3

Before Monday, Page 165-166 Introduction to Section 3.1.
Before Wednesday, Example 3.9 on page 173.
Before Friday, Page 184-185 Terminology for describing Turing machines.
Week 7 Monday: No class, in observance of Veterans Day. For Week 7 Wednesday: Introduction to Chapter 4.

We will be learning and practicing to:

- Clearly and unambiguously communicate computational ideas using appropriate formalism. Translate across levels of abstraction.
  - Use precise notation to formally define the state diagram of a Turing machine
  - Use clear English to describe computations of Turing machines informally.
    - **Motivate the definition of a Turing machine**
    - **Trace the computation of a Turing machine on given input**
    - **Describe the language recognized by a Turing machine**
    - **Determine if a Turing machine is a decider**
    - **Given an implementation-level description of a Turing machine**
    - **Use high-level descriptions to define and trace Turing machines**
    - **Apply dovetailing in high-level definitions of machines**
  - Give examples of sets that are recognizable and decidable (and prove that they are).
    - **State the definition of the class of recognizable languages**
    - **State the definition of the class of decidable languages**
    - **State the definition of the class of co-recognizable languages**
- Know, select and apply appropriate computing knowledge and problem-solving techniques. Reason about computation and systems.
- Describe and prove closure properties of classes of languages under certain operations.
  - **Apply a general construction to create a new Turing machine from an example one.**
  - **Formalize a general construction from an informal description of it.**
  - **Use general constructions to prove closure properties of the class of decidable langugages.**
  - **Use general constructions to prove closure properties of the class of recognizable langugages.**

TODO:

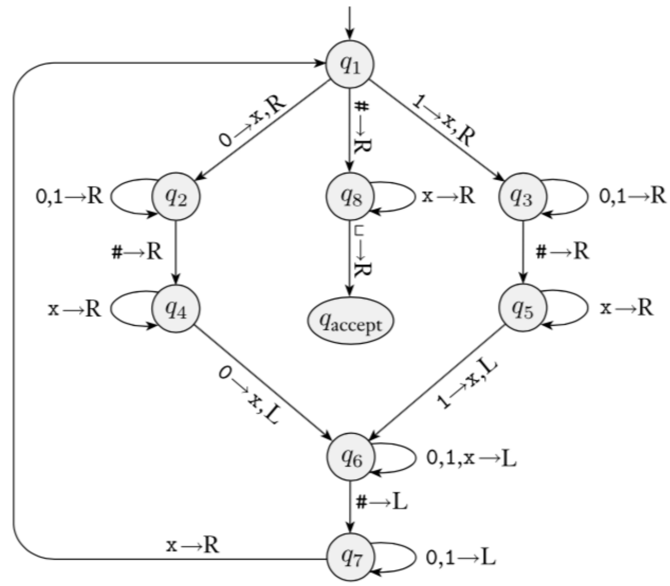
Review Quiz 6 on PrairieLearn (http://us.prairielearn.com), complete by Sunday 11/11/2024

Homework 4 submitted via Gradescope (https://www.gradescope.com/), due Tuesday 10/12/2024

Monday: Descriptions of Turing machines

Sipser Figure 3.10

Conventions in state diagram of TM: $b \to R$ label means $b \to b, R$ and all arrows missing from diagram represent transitions with output $(q_{reject}, ␣, R)$



Computation on input string 01#01

$q_1 \downarrow$						
0	1	#	0	1	␣	␣

Implementation level description of this machine:

> Zig-zag across tape to corresponding positions on either side of # to check whether the characters in these positions agree. If they do not, or if there is no #, reject. If they do, cross them off.
>
> Once all symbols to the left of the # are crossed off, check for any un-crossed-off symbols to the right of #; if there are any, reject; if there aren't, accept.

The language recognized by this machine is

$$\{w\#w \mid w \in \{0,1\}^*\}$$

High-level description of this machine is

Recall: High-level descriptions of Turing machine algorithms are written as indented text within quotation marks. Stages of the algorithm are typically numbered consecutively. The first line specifies the input to the machine, which must be a string.

Extra practice

Computation on input string 01#1

$q_1 \downarrow$						
0	1	#	1	␣	␣	␣

A language L is **recognized by** a Turing machine M means

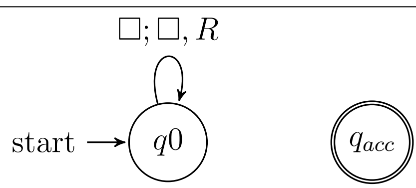
A Turing machine M **recognizes** a language L means

A Turing machine M is a **decider** means

A language L is **decided by** a Turing machine M means

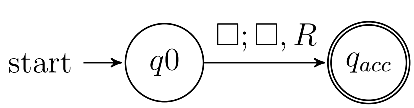
A Turing machine M **decides** a language L means

Fix $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \textvisiblespace\}$ for the Turing machines with the following state diagrams:

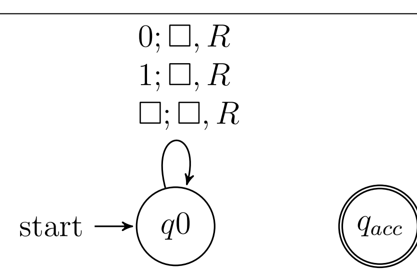


Decider? Yes / No

Decider? Yes / No



Decider? Yes / No



Decider? Yes / No

Wednesday: Recognizable and decidable languages

A **Turing-recognizable** language is a set of strings that is the language recognized by some Turing machine. We also say that such languages are recognizable.

A **Turing-decidable** language is a set of strings that is the language recognized by some decider. We also say that such languages are decidable.

An **unrecognizable** language is a language that is not Turing-recognizable.

An **undecidable** language is a language that is not Turing-decidable.

True or **False**: Any decidable language is also recognizable.

True or **False**: Any recognizable language is also decidable.

True or **False**: Any undecidable language is also unrecognizable.

True or **False**: Any unrecognizable language is also undecidable.

True or **False**: The class of Turing-decidable languages is closed under complementation.

Using formal definition:

Using high-level description:

Church-Turing Thesis (Sipser p. 183): The informal notion of algorithm is formalized completely and correctly by the formal definition of a Turing machine. In other words: all reasonably expressive models of computation are equally expressive with the standard Turing machine.

Friday: Closure for the classes of recognizable and decidable languages

Definition: A language L over an alphabet Σ is called **co-recognizable** if its complement, defined as $\Sigma^* \setminus L = \{x \in \Sigma^* \mid x \notin L\}$, is Turing-recognizable.

Theorem (Sipser Theorem 4.22): A language is Turing-decidable if and only if both it and its complement are Turing-recognizable.

Proof, first direction: Suppose language L is Turing-decidable. WTS that both it and its complement are Turing-recognizable.

Proof, second direction: Suppose language L is Turing-recognizable, and so is its complement. WTS that L is Turing-decidable.

Notation: The complement of a set X is denoted with a superscript c, X^c, or an overline, $\overline{X}$.

Claim: If two languages (over a fixed alphabet Σ) are Turing-decidable, then their union is as well.

Proof:

Claim: If two languages (over a fixed alphabet Σ) are Turing-recognizable, then their union is as well.

Proof:

