Lecture 8 - The Halting Problem

Eric A. Autry

Last time: Undecidability and the Church-Turing Thesis

This time: The Halting Problem

Next time: MIDTERM

Homework 3 is due **today.** Solutions will be posted Friday at 6 pm (all late homework due by then).

Midterm is **next Tuesday the 2nd.**

Undecidable Languages

Theorem

Some languages cannot be recognized by a Turing Machine.

Proof:

There are a countable number of Turing Machines, but an uncountable number of languages!

Church-Turing Thesis

Since 1936, many variations of machines have been proposed, but all have ended up being equivalent to (or less powerful than) a Turing Machine.

This lead to the Church-Turing Thesis:

- It basically states that the intuitive notion of an algorithm is equivalent to Turing Machine algorithms.

i.e.,

- If a solution to a problem is calculable, a Turing Machine can compute it.

This hasn't been proven, but so far nobody has come up with a better machine...

Proving Undecidability

So, there exist undecidable problems...

How do we show if a problem is undecidable?

We can either prove it by directly considering the problem,

- Remember that we proved that A_{TM} is undecidable:

$$A_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts input string } w\}$$

or **reduce** the problem to another undecidable problem.

Famous undecidable problem used in many proofs: The Halting Problem.

The Halting Problem

$$HALT_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ halts on input } w\}$$

Theorem

$HALT_{TM}$ is undecidable.

Proof:

Before we start, note that we can think of these machines as just programs written in some sort of TM language.

The Halting Problem

$$HALT_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ halts on input } w\}$$

Theorem

$HALT_{TM}$ is undecidable.

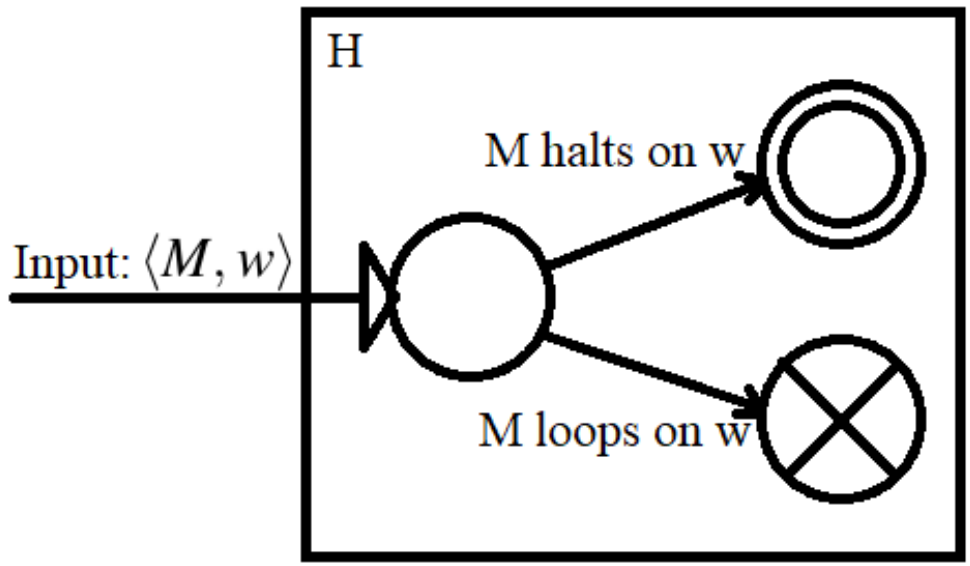
Proof:

Assume by way of contradiction that $HALT_{TM}$ is decidable. Then we have machine H that decides it, i.e., a machine H that will always halt and either accept or reject.

$$H(\langle M, w\rangle) = \begin{cases} accept & \text{if } M \text{ halts on input } w, \\ reject & \text{if } M \text{ loops on input } w \end{cases}$$

The Halting Problem

$$H(\langle M, w \rangle) = \begin{cases} accept & \text{if } M \text{ halts on input } w, \\ reject & \text{if } M \text{ loops on input } w \end{cases}$$

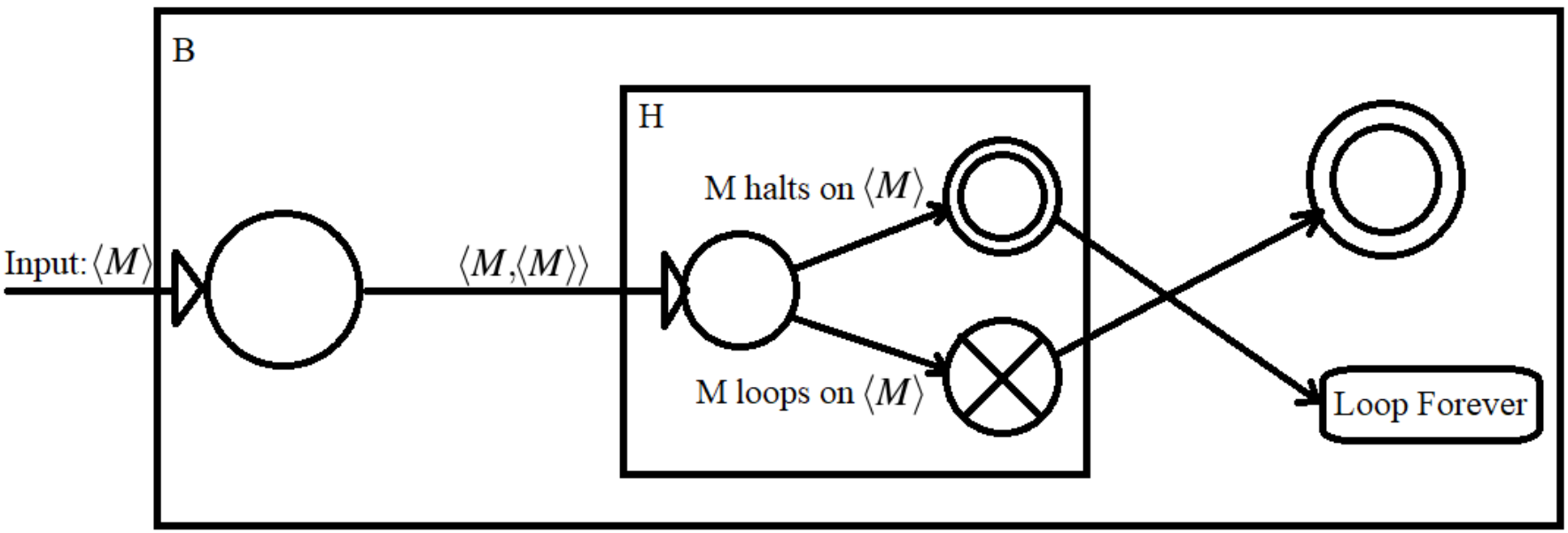


Note that we can input the string $\langle M \rangle$ into the machine M and ask if it halts: $H(\langle M, \langle M \rangle \rangle)$.

The Halting Problem

Let's now build a new machine B:

$$B(\langle M \rangle) = \begin{cases} accept & \text{if } H \text{ rejects (i.e., if } M \text{ will loop on } \langle M \rangle), \\ \textbf{\textit{loop}} & \text{if } H \text{ accepts (i.e., if } M \text{ will halt on } \langle M \rangle), \end{cases}$$



What happens if we run B on itself? Does B halt on input $\langle B \rangle$?

The Halting Problem

What happens if we run B on itself? Does B halt on input $\langle B \rangle$?

$$B(\langle B \rangle) = \begin{cases} accept & \text{if } H \text{ rejects, telling us that } B \text{ will loop on } \langle B \rangle, \\ \textbf{\textit{loop}} & \text{if } H \text{ accepts, telling us that } B \text{ will halt on } \langle B \rangle, \end{cases}$$

But wait, we are currently running B on input $\langle B \rangle$...

- If H says that B will loop, then B halts instead!
- If H says that B will halt, then B loops instead!

Other Undecidable Languages

- Will a given machine M accept no inputs at all, i.e., $L(M) = \varnothing$?
- Do two Turing Machines M_1 and M_2 accept the same language, i.e., $L(M_1) = L(M_2)$?
- Can a given context free grammar G generate all strings, i.e., $L(G) = \Sigma^*$?

Rice's Theorem:
Any nontrivial property about the language recognized by a Turing machine is undecidable.

These are all questions about machines themselves. It turns out that some simple questions not related to machines are also undecidable.
(Ex: Post Correspondence Problem)

Midterm Review

Very Important Midterm Announcement:

This exam must be individual work.

You may not collaborate with your fellow students.

However, you may use your notebook, which must contain only handwritten notes, written by you.

Please aim to be a couple minutes early to the midterm.

Midterm Study Material

The main study material for the midterm will be:

Course notes0 - notes6 on Sakai.

Homework solutions on Sakai.

This review in notes8 on Sakai.

You may assume that anything we proved in class or in the homework is a theorem you may use on the exam.

Auditing the Midterm

If you are auditing the course, I will allow you to take the midterm only if you have completed quiz 0 on Sakai.

If you choose to take the midterm, your exams will be collected separately at the end of class and will not be graded.

Sets - Notation

A set of elements a, b, and c is written as

$$\mathcal{S} = \{a, b, c\}.$$

To indicate that a is in set $\mathcal{S}$, we saw 'a is an element of set $\mathcal{S}$', and write

$$a \in \mathcal{S}.$$

We say that the sets A and B are the **same size** if there exists a function $f : A \to B$ that is one-to-one and onto. We call this function a **correspondence**.

Countable Sets

A set is **countable** if it either is finite or has the same size as $\mathbb{N}$.

This means that a set is countable if we can assign the natural numbers to the elements of the set.

i.e., if we can **count** the elements of the set.

New Vocab: When we call a set finite, countable, or uncountable, we are referring to what is called the set's **cardinality,** another way of referring to the size of the set.

Sets - Named Sets

- Natural numbers: $\mathbb{N} = \{1, 2, 3, 4, \dots\}$ (countable)
- Integers: $\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$
  (**Practice problem...** countable)
- Rational numbers: $\mathbb{Q} = \{x |\ x = a/b$ where $a, b \in \mathbb{Z}\}$
  (countable)
- Real numbers: $\mathbb{R}$ (uncountable)
- Empty (Null) Set: $\varnothing = \{\}$

Sets - Operations

Common Set Operations:

- Union:
  $A \cup B$ is the set of all things that are in either A or B.
- Intersection:
  $A \cap B$ is the set of all things that are in both A and B.
- Complement:
- $\overline{A}$ is the set of all things that are not in A.

Note that union and intersection can be defined in terms of each other:

- $A \cup B = \overline{\left(\overline{A} \cap \overline{B}\right)}$.
- $A \cap B = \overline{\left(\overline{A} \cup \overline{B}\right)}$.

Sets - Subsets

- Subset:
  $A \subseteq B$, every element in set A is also in set B.

- Proper Subset:
  $A \subset B$, set A is a subset of set B and $A \neq B$.

Note that a subset of a countable set is itself always countable.

- Since we have a listing for the countable set, we can simply erase the elements from the list that are not in the subset, and shift all of the other elements up in the list.
- i.e., since the subset cannot be larger than the original set, the subset cannot be uncountable while the original set is countable.

Sequences and Tuples

A sequence is a list of objects in some order.

- Ex: (3,65,2) or (65,3,2). Note: these are not the same sequence because order matters.
- Note: a sequence can be infinite.

A tuple is a sequence with a finite length, and a tuple of length k is called a k-tuple.

- Ex: (3,65,2) is a 3-tuple and (5,4) is a 2-tuple.
- 2-tuples are also called ordered pairs.

Sets, Sequences, and Tuples

The power set $\mathcal{P}(A)$ of set A is the set of all subsets of A.

If $A = \{0, 1\}$, then the power set of A is the set:

$$\{ \varnothing, \{0\}, \{1\}, \{0, 1\} \}.$$

Note that the power set $\mathcal{P}(A)$ of a countable set A is uncountable.

This is because the power set includes the infinite subsets.

Practice problem... *Proof:* there is a correspondence between A and $\mathbb{N}$, but also a correspondence of $\mathcal{P}(A)$ and the set of all subsets of $\mathbb{N}$, which you proved was uncountable in your homework.

Strings

We define an alphabet as a nonempty finite set of symbols. We typically use the letters Σ and Γ to denote alphabets.

- $\Sigma = \{0, 1\}$
- $\Gamma = \{a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z\}$

A string over an alphabet is a finite sequence of symbols from that alphabet.

- $\Sigma = \{0, 1\}$,
  Ex: 0011, 01001, 0, 10
- The empty string is a string of length 0, denoted ε.

Properties:

- The length of a string w is written $|w|$.
- The reverse of a string w is written $w^{\mathcal{R}}$.
- A string z is a substring of w if z appears within w.
  Ex: ‘abba’ is a substring of ‘bbabba’

Languages

A language is a set of strings.

We know a lot more about these now...

Regular Languages - Review

Regular languages can be recognized (and decided) by DFAs, NFAs, and Regular Expressions.

NFAs can be reduced to DFAs, and DFAs can be reduced to Regular Expressions.

So all three are equivalent in power and for the class of languages called regular.

Regular languages are closed under the union, concatenation, star, and reverser operations (what others?)

Regular Languages - Potential Problems

- DFAs
  - What does this DFA do?
  - Write a DFA for this language.
- NFAs
  - What does this NFA do?
  - Write a NFA for this language using k states.
  - Reduce this NFA to a DFA.
  - Reduce this NFA to a Regular Expression
- Regular Expressions
  - Write down a regular expression for this language.
- Regular Language Proofs
  - Prove that regular languages are closed under this operation.
  - i.e., describe a procedure to convert a group of given NFAs into a new NFA that can recognize the operation.
  - For such a procedure, you will need at least a sentence justifying why your new machine does the correct thing (see HW 2 solutions for example on reverse).

Context Free Languages - Last Mention

We made more powerful machines called PDAs by adding an infinite stack.

They are equivalent to CFGs.

These correspond to the Context Free Languages.

Not on the exam.

Turing Machines - Review

We build the more powerful Turing Machines that have an infinite tape.

This tape corresponds to something like disk memory, but without addressing (i.e., the only way you can read a symbol is by stepping to it one place at a time).

Turing Machines can decide (always halt) or semidecide a language.

We describe algorithms for Turing Machines.

Turing Machines - Review

Church-Turing Thesis: Turing Machines are the most powerful.

We can think of Turing Machines as written programs in the form of the algorithms we describe (instead of python or C, we now have the language TM).

Turing Machines - Potential Problems

Describe a Turing Machine that can decide a given language.

You must provide a reasonable level of detail, but do not have to give info on actual states of the machine.

You may assume that the machines can find the first symbol or the last symbol, and can compare two separated strings.

The machine can mark the tape.

If you want the machine to do something more complicated, you must tell me how the machine would accomplish that task.

- Ex: finding the length of a string. You would need to explain how the machine does the counting and how the machine stores that info (note: storing the length has to be done on the tape).
- You will never be *required* to use a technique like this (although you can if you take the time to describe how).

Countable Sets - Potential Problems

I could ask you to prove that a set is countable.

- You can describe a way to list/count the set, which means that you have found a way to uniquely assign the natural numbers to elements of your countable set.
- You can provide a function that is a correspondence
  Ex: $f(n) = 2n$

Proofs require a reasonable amount of details (enough to be convincing). When you provide a way to list/count the set, you will need at least one sentence justifying why your system works.

Countable Sets - Potential Problems

For example: you could conclude something like “because each of the diagonals has a finite number of rationals, and there is a countable number of diagonals (1st diagonal, 2nd diagonal, etc), we can conclude that there are a countable number of rationals in total.”

You may now take for given that a countable set of finite elements contains a countable number of those elements in total.

You may take for given that a subset of a countable set is also countable.

Uncountable Sets - Potential Problems

Not on the exam.

Very Important Midterm Announcement:

This exam must be individual work.

You may not collaborate with your fellow students.

However, you may use your notebook, which must contain only handwritten notes, written by you.

Please aim to be a couple minutes early to the midterm.

Midterm Study Material

The main study material for the midterm will be:

Course notes0 - notes6 on Sakai.

Homework solutions on Sakai.

This review in notes8 on Sakai.

You may assume that anything we proved in class or in the homework is a theorem you may use on the exam.

Auditing the Midterm

If you are auditing the course, I will allow you to take the midterm only if you have completed quiz 0 on Sakai.

If you choose to take the midterm, your exams will be collected separately at the end of class and will not be graded.