

FIT2014 Theory of Computation

# Assignment Project Exam Help

Lecture 22

Undecidability

<https://powcoder.com>

slides by Graham Farr

## Add WeChat powcoder

COMMONWEALTH OF AUSTRALIA

Copyright Regulations 1969

Warning

This material has been reproduced and communicated to you by or on behalf of Monash University  
in accordance with s113P of the Copyright Act 1968 (the Act).

The material in this communication may be subject to copyright under the Act.

Any further reproduction or communication of this material by you may be the subject of copyright protection under the Act.

Do not remove this notice.

# Assignment Project Exam Help

- ▶ Halting Problem (or Entscheidungsproblem)
- ▶ Proof of its undecidability
- ▶ Using mapping reductions to prove undecidability
- ▶ Other undecidable problems

<https://powcoder.com>

Add WeChat powcoder

## Undecidable languages exist

The set of all deciders is countable.

- ▶  $\{\text{CWL-encodings of deciders}\} \subseteq \{\text{CWL}\} \subseteq \Sigma^*$
- ▶ and  $\Sigma^*$  is countable. (Lecture 5)

The set of all decidable languages is countable.

The set of *all* languages is uncountable. (Lecture 5)

Therefore undecidable languages exist.

## Halting Problem: Definition

### Halting Problem

INPUT: Turing machine  $P$ , input  $x$

QUESTION: If  $P$  is run with input  $x$  does it eventually halt?

As a language:

HaltingProblem  $:= \{ \langle P, x \rangle : \text{when } P \text{ is run with input } x, \text{ it eventually halts.} \}$

### Theorem.

The Halting Problem is undecidable

Proved by:

- ▶ Alonzo Church (1936): lambda calculus
- ▶ Alan Turing (1936-37): Turing machines

## Halting Problem

### Theorem

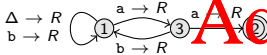
The Halting Problem is undecidable.

Proof ingredients:

- ▶ contradiction
- ▶ diagonalisation
- ▶ a version of the Liar Paradox: "This sentence is false."

Consider what happens when we run Turing machines (encoded as strings) on input strings.

✓ = Halts;    ✗ = Doesn't halt.

		inputs to TMs									
		$\epsilon$	a	b	aa	ab	ba	bb	aaa	aab	...
Turing machines  $\Delta \rightarrow R$ $b \rightarrow R$ $a \rightarrow R$ $a \rightarrow R$ $b \rightarrow R$ ababababbababbbbbbabaaabaaaab- aaababababbbaaabaabaaaab	$\epsilon$	✓	✗	✗	✓	✗	✓	✓	✓	✗	...
	a	✗	✗	✓	✓	✗	✗	✗	✓	✓	...
	b	✓	✓	✗	✗	✗	✓	✗	✗	✓	...
	aa	✓	✗	✓	✓	✗	✗	✓	✓	✗	...
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...

E:    ✗    ✓    ✓    ✗    ...

## Halting Problem is Undecidable

**Proof.** (by contradiction)

# Assignment Project Exam Help

Assume there is a Decider,  $D$ , for the Halting Problem.

So it can tell, for any  $P$  and  $x$ , whether or not  $P$  eventually halts after being given input  $x$ .

# <https://powcoder.com>

So it can tell, for any  $P$ , whether or not  $P$  eventually halts after being given input  $P$ !

# Add WeChat powcoder

Construct another program (Turing machine)  $E$  as follows ...

## Halting Problem is Undecidable (cont'd)

$E$

Input:  $P$

Use  $D$  to determine what happens if  $P$  runs on itself.

If  $D$  says, " $P$  halts, with input  $P$ ": loop forever.

If  $D$  says, " $P$  loops forever, with input  $P$ ": Halt.

What happens when  $E$  is given itself as input?

If  $E$  halts, for input  $E$ : then  $E$  loops forever, for input  $E$ .

If  $E$  loops forever, for input  $E$ : then  $E$  halts, for input  $E$ .

Contradiction!



YouTube film of proof: <https://www.youtube.com/watch?v=92WHN-pAFCs>



## Other Undecidable Problems

### DIAGONAL HALTING PROBLEM

INPUT: Turing machine  $P$

QUESTION: Does  $P$  eventually halt, for input  $P$ ?

Above proof already shows this.

### HALT FOR INPUT ZERO

INPUT: Turing machine  $P$

QUESTION: Does  $P$  eventually halt, for input 0?

### Theorem.

HALT FOR INPUT ZERO is undecidable.

We'll prove this by mapping reduction from the Diagonal Halting Problem.

# Assignment Project Exam Help

Recall:

If there is a mapping reduction  $f$  from  $K$  to  $L$ , then:

<https://powcoder.com>

If  $L$  is decidable, then  $K$  is decidable.

If  $K$  is undecidable, then  $L$  is undecidable.

Add WeChat powcoder

**Proof.** ... that HALT FOR INPUT ZERO is undecidable:

Let  $M$  be any program, which we regard as an input to the Diagonal Halting Problem.  
Define  $M'$  as follows:

$M'$

Input:  $x$   
Run  $M$  on input  $M$ .

Observe:

- ▶ The construction  $M \mapsto M'$  is computable.
- ▶  $M$  halts on input  $M$  if and only if  $M'$  halts on input  $x$ .

So, the function that sends  $M \mapsto M'$  is a mapping reduction from  
DIAGONAL HALTING PROBLEM to HALT FOR INPUT ZERO.

Therefore HALT FOR INPUT ZERO is undecidable.

## Other Undecidable Problems

There's nothing special about zero, here.

So we get a whole lot of undecidability results

For example:

HALT FOR INPUT 42

INPUT: Turing machine  $P$

QUESTION: Does  $P$  eventually halt, for input 42?

Proof of undecidability is virtually identical to the previous one ...

Use a mapping reduction, with 42 instead of 0.

# Assignment Project Exam Help

ALWAYS HALTS

INPUT: Turing machine  $P$

QUESTION: Does  $P$  always halt eventually, for any input?

**Theorem.**

ALWAYS HALTS is undecidable.

Proof is virtually identical to the previous one...

<https://powcoder.com>

Add WeChat powcoder

**Proof.** ... that ALWAYS HALTS is undecidable:

Let  $M$  be any program, which we regard as an input to the Diagonal Halting Problem.  
Define  $M'$  as follows:

$M'$

Input:  $x$   
Run  $M$  on input  $M$ .

Observe:

- ▶ The construction  $M \mapsto M'$  is computable.
- ▶  $M$  halts on input  $M$  if and only if  $M'$  always halts.

So, the function that sends  $M \mapsto M'$  is a mapping reduction from  
DIAGONAL HALTING PROBLEM to ALWAYS HALTS.

Therefore ALWAYS HALTS is undecidable.

# SOMETIMES HALTS Assignment Project Exam Help

INPUT: Turing machine  $P$

QUESTION: Is there some input for which  $P$  eventually halts?

**Theorem.**

SOMETIMES HALTS is undecidable.

Proof is virtually identical to the previous one . . .

<https://powcoder.com>

Add WeChat powcoder

**Proof.** ... that **SOMETIMES HALTS** is undecidable:

Let  $M$  be any program, which we regard as an input to the Diagonal Halting Problem.  
Define  $M'$  as follows:

$M'$

Input:  $x$   
Run  $M$  on input  $M$ .

Observe:

- ▶ The construction  $M \mapsto M'$  is computable.
- ▶  $M$  halts on input  $M$  if and only if  $M'$  halts for some input.

So, the function that sends  $M \mapsto M'$  is a mapping reduction from  
DIAGONAL HALTING PROBLEM to **SOMETIMES HALTS**.

Therefore **SOMETIMES HALTS** is undecidable.



## Other Undecidable Problems

NEVER HALTS

INPUT: Turing machine  $P$

QUESTION: Does  $P$  always loop forever for any input?

**Theorem.**

NEVER HALTS is undecidable.

**Proof.** by a more general type of reduction, from SOMETIMES HALTS.

If  $D$  is a decider for NEVER HALTS, then switching Accept and Reject gives a decider for SOMETIMES HALTS.

But we now know that SOMETIMES HALTS is undecidable.

Contradiction.

So NEVER HALTS is undecidable too.



## Other Undecidable Problems

INPUT: Turing machine  $P$  and  $Q$

QUESTION: Do  $P$  and  $Q$  always both halt, or both loop?

i.e., is it the case that:

$\forall x: P \text{ halts on input } x \iff Q \text{ halts on input } x \dots?$

INPUT: Turing machine  $P$

QUESTION: If  $P$  is run on the input "What's the answer?", does it output "42"?

## Decidable or Undecidable?

INPUT: Turing machine  $P$ , input  $x$ .

QUESTION: Does  $P$  accept  $x$ ?

INPUT: Turing machine  $P$ , input  $x$ , positive integer  $t$ .

QUESTION: When  $P$  is run on  $x$ , does it halt in  $\leq t$  steps?

INPUT: Turing machine  $P$ , positive integer  $s$ .

QUESTION: Does  $P$  have  $\leq s$  states?

INPUT: Turing machine  $P$ , positive integer  $k$ .

QUESTION: Does  $P$  halt for some input of length  $\leq k$ .

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

## Other Undecidable Problems

INPUT: a Turing machine  $P$

QUESTION: Is  $\text{Accept}(P)$  regular?

i.e., is  $P$  equivalent to a Finite Automaton?

INPUT: a CFG

QUESTION: is the language it generates regular?

INPUT: a CFG

QUESTION: is there any string that it doesn't generate? (over same alphabet)

INPUT: two CFGs.

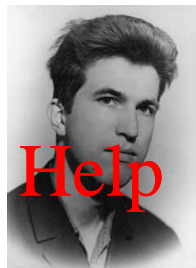
QUESTION: Do they define the same language?

## Other Undecidable Problems

INPUT: a polynomial (in several variables)

QUESTION: Does it have an integer root?

(Y. Matiyasevich, 1970)



<https://mathshistory.st-andrews.ac.uk/Biographies/Matiyasevich/>

Yuri Matiyasevich (b. 1947)

<https://powcoder.com>

Post Correspondence Problem

(a problem about string matching;  
see Sipser, Section 5.2)



<https://mathshistory.st-andrews.ac.uk/Biographies/Post/>

Emil Post (1897–1954)

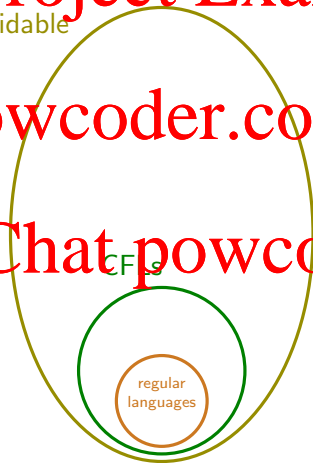
Add WeChat powcoder

Assignment Project Exam Help

Decidable

<https://powcoder.com>

Add WeChat powcoder



- # Assignment Project Exam Help
- ▶ Know and understand the Halting Problem
  - ▶ Prove its undecidability
  - ▶ Be able to use mapping reductions to prove undecidability
  - ▶ Know examples of undecidable problems.

Reading: Sipser, pp. 201–209, 215–220, 234–236.

Preparation: Sipser, pp. 170, 209–210.