Theory of Computer Science

C4. Reductions

Gabriele Röger

University of Basel

April 23, 2025

Introduction

What We Achieved So Far: Discussion

- We already know a concrete undecidable problem.
  $\rightarrow$ halting problem
- We will see that we can derive further undecidability results from the undecidability of the halting problem.
- The central notion for this is reducing one problem to another problem.

Illustration

```
def is_odd(some_number):
    n = some_number + 1
    return is_even(n)
```

- Decides whether a given number is odd based on...
- an algorithm that determines whether a number is even.

Reduction: Idea (slido)

Assume that you have an algorithm that solves problem A relying on a hypothetical algorithm for problem B.

```
def is_in_A(input_A):
  input_B = <compute suitable instance based on input_A>
  return is_in_B(input_B)
```

Reduction: Idea (slido)

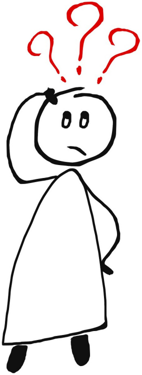
Assume that you have an algorithm that solves problem A relying on a hypothetical algorithm for problem B.

```
def is_in_A(input_A):
  input_B = <compute suitable instance based on input_A>
  return is_in_B(input_B)
```

What (if anything) can you conclude

1. if there indeed is an algorithm for problem A?
2. if there indeed is an algorithm for problem B?
3. if problem A is undecidable?
4. if problem B is undecidable?

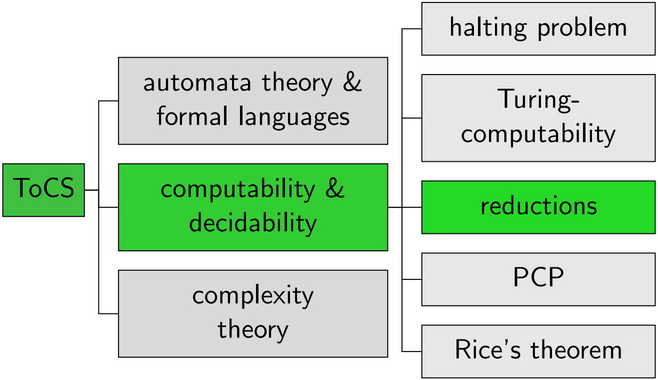
Questions



Questions?

Reduction

Content of the Course



Reduction: Definition

Definition (Reduction)

Let $A \subseteq \Sigma^*$ and $B \subseteq \Gamma^*$ be languages, and let $f : \Sigma^* \to \Gamma^*$ be a total and computable function such that for all $x \in \Sigma^*$:

$$x \in A \quad \text{if and only if} \quad f(x) \in B.$$

Then we say that A can be reduced to B (in symbols: $A \leq B$), and f is called a reduction from A to B.

German: A ist auf B reduzierbar, Reduktion von A auf B

Reduction Property

Theorem (Reductions vs. Turing-recognizability/Decidability)

Let A and B be languages with $A \leq B$. Then:

1. *If B is decidable, then A is decidable.*
2. *If B is Turing-recognizable, then A is Turing-recognizable.*
3. *If A is not decidable, then B is not decidable.*
4. *If A is not Turing-recognizable, then B is not Turing-recognizable.*

$\rightsquigarrow$ In the following, we use 3. to show undecidability for further problems.

Reduction Property: Proof

Proof.

for 1.: If B is decidable then there is a DTM M_B that decides B. The following algorithm decides A using reduction f from A to B.

On input x:

1. $y := f(x)$
2. Simulate M_B on input y. This simulation terminates.
3. If M_B accepted y, accept. Otherwise reject.

Reduction Property: Proof

Proof.

for 1.: If B is decidable then there is a DTM M_B that decides B. The following algorithm decides A using reduction f from A to B.

On input x:

1. $y := f(x)$
2. Simulate M_B on input y. This simulation terminates.
3. If M_B accepted y, accept. Otherwise reject.

for 2.: identical to (1), only that M_B only recognizes B and therefore the simulation does not necessarily terminate if $y \notin B$. Since $y \notin B$ iff $x \notin A$, the procedure still recognizes A.

Reduction Property: Proof

Proof.

for 1.: If B is decidable then there is a DTM M_B that decides B. The following algorithm decides A using reduction f from A to B.

On input x:

1. $y := f(x)$
2. Simulate M_B on input y. This simulation terminates.
3. If M_B accepted y, accept. Otherwise reject.

for 2.: identical to (1), only that M_B only recognizes B and therefore the simulation does not necessarily terminate if $y \notin B$. Since $y \notin B$ iff $x \notin A$, the procedure still recognizes A.

for 3./4.: contrapositions of 1./2. $\rightsquigarrow$ logically equivalent □

Reductions are Preorders

Theorem (Reductions are Preorders)

The relation "$\leq$" is a preorder:

1. *For all languages A:*
   $A \leq A$ *(reflexivity)*
2. *For all languages A, B, C:*
   If $A \leq B$ and $B \leq C$, then $A \leq C$ (transitivity)

German: schwache Halbordnung/Quasiordnung, Reflexivität, Transitivität

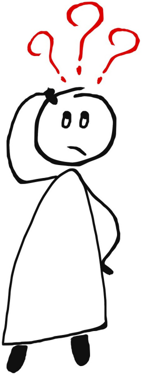
Reductions are Preorders: Proof

Proof.

for 1.: The function $f(x) = x$ is a reduction from A to A because it is total and computable and $x \in A$ iff $f(x) \in A$.

for 2.: $\rightsquigarrow$ exercises $\square$

Questions



Questions?

Halting Problem on Empty Tape

Example

As an example

- we will consider problem H_0, a variant of the halting problem,
- . . . and show that it is undecidable
- . . . reducing H to H_0.

Reminder: Halting Problem

Definition (Halting Problem)

The halting problem is the language

$$H = \{w\#x \in \{0,1,\#\}^* \mid w, x \in \{0,1\}^*, \\ M_w \text{ started on } x \text{ terminates}\}$$

Halting Problem on Empty Tape (1)

Definition (Halting Problem on the Empty Tape)

The halting problem on the empty tape is the language

$$H_0 = \{w \in \{0,1\}^* \mid M_w \text{ started on } \varepsilon \text{ terminates}\}.$$

German: Halteproblem auf leerem Band

Halting Problem on Empty Tape (1)

Definition (Halting Problem on the Empty Tape)

The halting problem on the empty tape is the language

$$H_0 = \{w \in \{0,1\}^* \mid M_w \text{ started on } \varepsilon \text{ terminates}\}.$$

Note: H_0 is Turing-recognizable. (Why?)

German: Halteproblem auf leerem Band

Halting Problem on Empty Tape (2)

Theorem (Undecidability of Halting Problem on Empty Tape)

The halting problem on the empty tape is undecidable.

Halting Problem on Empty Tape (2)

Theorem (Undecidability of Halting Problem on Empty Tape)

The halting problem on the empty tape is undecidable.

Proof.

We show $H \leq H_0$.

Consider the function $f : \{0, 1, \#\}^* \to \{0, 1\}^*$
that computes the word $f(z)$ for a given $z \in \{0, 1, \#\}^*$ as follows:

Halting Problem on Empty Tape (2)

Theorem (Undecidability of Halting Problem on Empty Tape)

The halting problem on the empty tape is undecidable.

Proof.

We show $H \leq H_0$.

Consider the function $f : \{0, 1, \#\}^* \to \{0, 1\}^*$
that computes the word $f(z)$ for a given $z \in \{0, 1, \#\}^*$ as follows:

- Test if z has the form $w\#x$ with $w, x \in \{0, 1\}^*$.
- If not, return any word that is not in H_0
  (e. g., encoding of a TM that instantly starts an endless loop).
- If yes, split z into w and x.

Halting Problem on Empty Tape (2)

Theorem (Undecidability of Halting Problem on Empty Tape)

The halting problem on the empty tape is undecidable.

Proof.

We show $H \leq H_0$.

Consider the function $f : \{0, 1, \#\}^* \to \{0, 1\}^*$
that computes the word $f(z)$ for a given $z \in \{0, 1, \#\}^*$ as follows:

- Test if z has the form $w\#x$ with $w, x \in \{0, 1\}^*$.
- If not, return any word that is not in H_0 (e. g., encoding of a TM that instantly starts an endless loop).
- If yes, split z into w and x.
- Decode w to a TM M_2.

...

Halting Problem on Empty Tape (3)

Proof (continued).

- Construct a TM M_1 that behaves as follows:
  - If the input is empty: write x onto the tape and move the head to the first symbol of x (if $x \neq \varepsilon$); then stop
  - otherwise, stop immediately

Halting Problem on Empty Tape (3)

Proof (continued).

- Construct a TM M_1 that behaves as follows:
  - If the input is empty: write x onto the tape and move the head to the first symbol of x (if $x \neq \varepsilon$); then stop
  - otherwise, stop immediately
- Construct TM M that first runs M_1 and then M_2.
  $\rightarrow$ M started on empty tape simulates M_2 on input x.

Halting Problem on Empty Tape (3)

Proof (continued).

- Construct a TM M_1 that behaves as follows:
  - If the input is empty: write x onto the tape and move the head to the first symbol of x (if $x \neq \varepsilon$); then stop
  - otherwise, stop immediately
- Construct TM M that first runs M_1 and then M_2.
  $\rightarrow$ M started on empty tape simulates M_2 on input x.
- Return the encoding of M.

Halting Problem on Empty Tape (3)

Proof (continued).

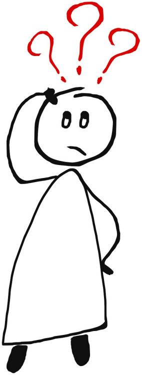
- Construct a TM M_1 that behaves as follows:
  - If the input is empty: write x onto the tape and move the head to the first symbol of x (if $x \neq \varepsilon$); then stop
  - otherwise, stop immediately
- Construct TM M that first runs M_1 and then M_2.
  $\rightarrow$ M started on empty tape simulates M_2 on input x.
- Return the encoding of M.

f is total and (with some effort) computable. Also:

$$\begin{aligned} z \in H &\text{ iff } z = w\#x \text{ and } M_w \text{ run on } x \text{ terminates} \\ &\text{ iff } M_{f(z)} \text{ started on empty tape terminates} \\ &\text{ iff } f(z) \in H_0 \end{aligned}$$

$\rightsquigarrow H \leq H_0 \rightsquigarrow H_0$ undecidable $\square$

Questions



Questions?

Summary

Summary

- reductions: "embedding" a problem as a special case of another problem
- important method for proving undecidability: reduce from a known undecidable problem to a new problem