Theory of Computer Science

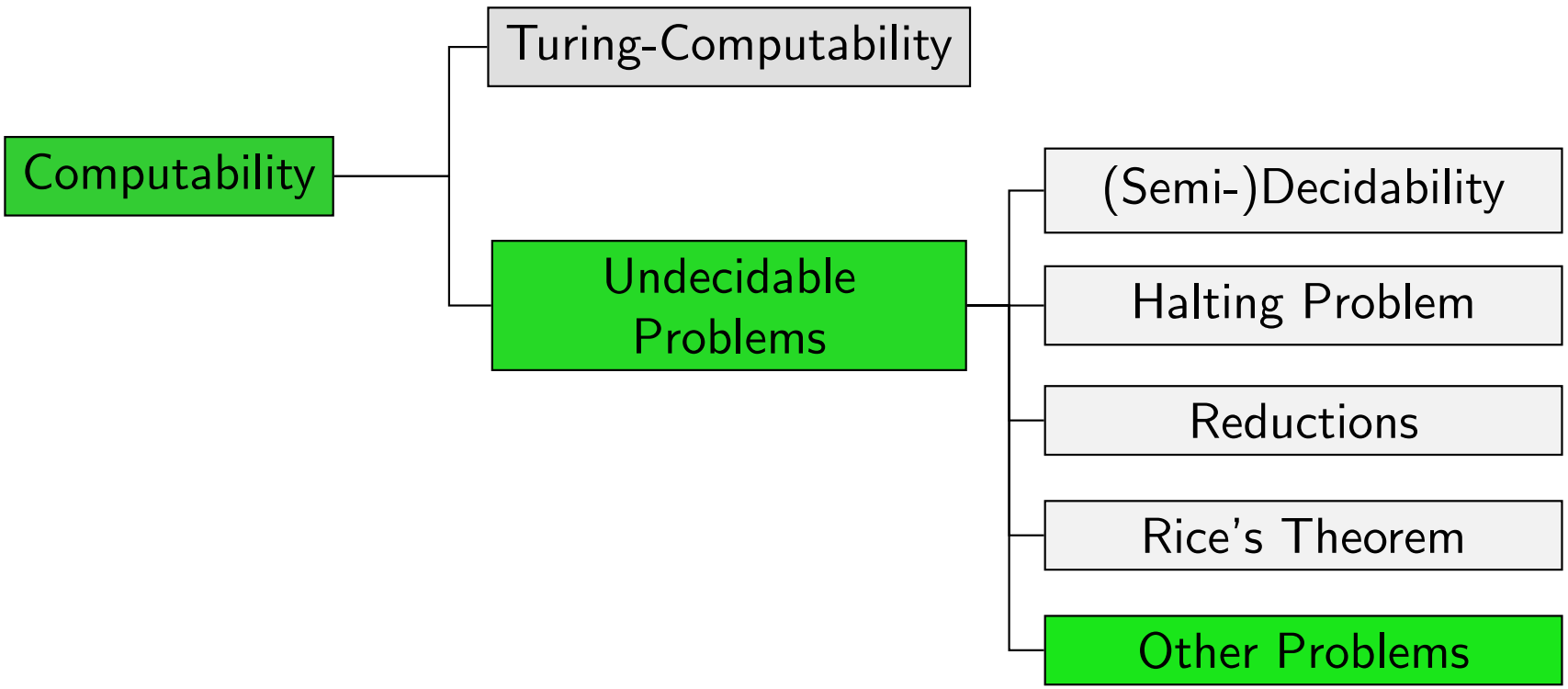
D5. Post Correspondence Problem

Gabriele Röger

University of Basel

May 4, 2020

Overview: Computability Theory



Post Correspondence Problem

How to prove undecidability?

- statements on the computed function of a TM/an algorithm
- other problems

How to prove undecidability?

- statements on the computed function of a TM/an algorithm
  $\rightarrow$ easiest with Rice' theorem
- other problems

How to prove undecidability?

- statements on the computed function of a TM/an algorithm
  $\rightarrow$ easiest with Rice' theorem
- other problems
  - directly with the definition of undecidability
    $\rightarrow$ usually quite complicated

How to prove undecidability?

- statements on the computed function of a TM/an algorithm
  $\rightarrow$ easiest with Rice' theorem
- other problems
  - directly with the definition of undecidability
    $\rightarrow$ usually quite complicated
  - reduction from an undecidable problem, e.g.
    $\rightarrow$ (general) halting problem (H)
    $\rightarrow$ halting problem on the empty tape (H_0)

More options for reduction proofs?

☹ all halting problems are quite similar

More options for reduction proofs?

☹ all halting problems are quite similar

→ We want a wider selection for reduction proofs
→ Is there some problem that is different in flavor?

More options for reduction proofs?

☹ all halting problems are quite similar

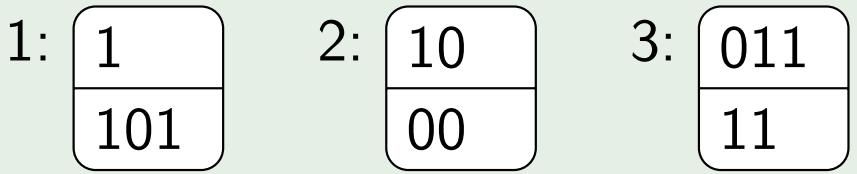
→ We want a wider selection for reduction proofs
→ Is there some problem that is different in flavor?

Post correspondence problem
(named after mathematician Emil Leon Post)

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'



(an infinite number of each kind)

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'

1: $\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ 2: $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ 3: $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

(an infinite number of each kind)

Question: Sequence of dominos such that

upper and lower row match (= are equal)

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'

1: $\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ 2: $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ 3: $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

(an infinite number of each kind)

Question: Sequence of dominos such that

upper and lower row match (= are equal)

$\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$

1

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'

1: $\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ 2: $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ 3: $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

(an infinite number of each kind)

Question: Sequence of dominos such that

upper and lower row match (= are equal)

$\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$

1

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'

1: $\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ 2: $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ 3: $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

(an infinite number of each kind)

Question: Sequence of dominos such that

upper and lower row match (= are equal)

$\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$

1

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'

1: $\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ 2: $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ 3: $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

(an infinite number of each kind)

Question: Sequence of dominos such that

upper and lower row match (= are equal)

$\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

1 3

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'

1: $\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ 2: $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ 3: $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

(an infinite number of each kind)

Question: Sequence of dominos such that

upper and lower row match (= are equal)

$\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

1 3

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'

1: $\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ 2: $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ 3: $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

(an infinite number of each kind)

Question: Sequence of dominos such that

upper and lower row match (= are equal)

$\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

1 3

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'

1: $\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ 2: $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ 3: $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

(an infinite number of each kind)

Question: Sequence of dominos such that

upper and lower row match (= are equal)

$\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

1 3

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'

1: $\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ 2: $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ 3: $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

(an infinite number of each kind)

Question: Sequence of dominos such that

upper and lower row match (= are equal)

$\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$ $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$

1 3 2

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'

1: $\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ 2: $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ 3: $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

(an infinite number of each kind)

Question: Sequence of dominos such that

upper and lower row match (= are equal)

$$\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}\ \begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}\ \begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}\ \begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$$

1 3 2 3

Post Correspondence Problem: Example

Example (Post Correspondence Problem)

Given: different kinds of "'dominos"'

1: $\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ 2: $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ 3: $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

(an infinite number of each kind)

Question: Sequence of dominos such that

upper and lower row match (= are equal)

$\begin{array}{|c|}\hline 1 \\ \hline 101 \\ \hline \end{array}$ $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$ $\begin{array}{|c|}\hline 10 \\ \hline 00 \\ \hline \end{array}$ $\begin{array}{|c|}\hline 011 \\ \hline 11 \\ \hline \end{array}$

1 3 2 3

Post Correspondence Problem: Definition

Definition (Post Correspondence Problem PCP)

Given: Finite sequence of pairs of words $(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)$, where $x_i, y_i \in \Sigma^+$ (for an arbitrary alphabet Σ)

Question: Is there a sequence $i_1, i_2, \ldots, i_n \in \{1, \ldots, k\}$, $n \geq 1$, with $x_{i_1} x_{i_2} \ldots x_{i_n} = y_{i_1} y_{i_2} \ldots y_{i_n}$?

A solution of the correspondence problem is such a sequence $i_1, \ldots, i_n$, which we call a match.

Given-Question Form vs. Definition as Set

So far: problems defined as sets

Now: definition in Given-Question form

Definition (new problem P)

Given: Instance $\mathcal{I}$

Question: Does $\mathcal{I}$ have a specific property?

Given-Question Form vs. Definition as Set

So far: problems defined as sets
Now: definition in Given-Question form

Definition (new problem P)

Given: Instance $\mathcal{I}$
Question: Does $\mathcal{I}$ have a specific property?

corresponds to definition

Definition (new problem P)

The problem P is the language
$\mathrm{P} = \{w \mid w \text{ encodes an instance } \mathcal{I} \text{ with the required property}\}$.

PCP Definition as Set

We can alternatively define PCP as follows:

Definition (Post Correspondence Problem PCP)

Das Post Correspondence Problem PCP is the set

$$\begin{aligned}\mathrm{PCP} = \{w \mid\ & w \text{ encodes a sequence of pairs of words} \\ & (x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k), \text{ for which there is a} \\ & \text{sequence } i_1, i_2, \ldots, i_n \in \{1, \ldots, k\} \\ & \text{such that } x_{i_1} x_{i_2} \ldots x_{i_n} = y_{i_1} y_{i_2} \ldots y_{i_n}\}.\end{aligned}$$

(Un-)Decidability of PCP

Post Correspondence Problem

PCP cannot be so hard, huh?

Post Correspondence Problem

PCP cannot be so hard, huh?

– Is it?

Post Correspondence Problem

PCP cannot be so hard, huh?

– Is it?

1101	0110	1
1	11	110

Formally: $K = ((1101, 1), (0110, 11), (1, 110))$

Post Correspondence Problem

PCP cannot be so hard, huh?

– Is it?

1101	0110	1
1	11	110

Formally: $K = ((1101, 1), (0110, 11), (1, 110))$

$\rightarrow$ Shortest match has length 252!

Post Correspondence Problem

PCP cannot be so hard, huh?

– Is it?

1101	0110	1
1	11	110

Formally: $K = ((1101, 1), (0110, 11), (1, 110))$

$\rightarrow$ Shortest match has length 252!

10	0	100
0	001	1

Formally: $K = ((10, 0), (0, 001), (100, 1))$

Post Correspondence Problem

PCP cannot be so hard, huh?

– Is it?

1101	0110	1
1	11	110

Formally: $K = ((1101, 1), (0110, 11), (1, 110))$
$\rightarrow$ Shortest match has length 252!

10	0	100
0	001	1

Formally: $K = ((10, 0), (0, 001), (100, 1))$
$\rightarrow$ Unsolvable

PCP: Semi-Decidability

Theorem (Semi-Decidability of PCP)

PCP *is semi-decidable.*

PCP: Semi-Decidability

Theorem (Semi-Decidability of PCP)

PCP *is semi-decidable.*

Proof.

Semi-decision procedure for input w:

- If w encodes a sequence $(x_1, y_1), \ldots, (x_k, y_k)$ of pairs of words:
  Test systematically longer and longer sequences $i_1, i_2, \ldots, i_n$ whether they represent a match.
  If yes, terminate and return "yes".
- If w does not encode such a sequence: enter an infinite loop.

If $w \in \mathrm{PCP}$ then the procedure terminates with "yes",
otherwise it does not terminate. □

PCP: Undecidability

Theorem (Undecidability of PCP)

PCP *is undecidable*.

PCP: Undecidability

Theorem (Undecidability of PCP)

PCP *is undecidable*.

Proof via an intermediate other problem
modified PCP (MPCP)

1. Reduce MPCP to PCP (MPCP $\leq$ PCP)
2. Reduce halting problem to MPCP ($H \leq$ MPCP)

PCP: Undecidability

Theorem (Undecidability of PCP)

PCP *is undecidable.*

Proof via an intermediate other problem
modified PCP (MPCP)

1. Reduce MPCP to PCP (MPCP $\leq$ PCP)
2. Reduce halting problem to MPCP ($H \leq$ MPCP)

$\rightarrow$ Let's get started...

MPCP: Definition

Definition (Modified Post Correspondence Problem MPCP)

Given: Sequence of word pairs as for PCP

Question: Is there a match $i_1, i_2, \ldots, i_n \in \{1, \ldots, k\}$ with $i_1 = 1$?

Reducibility of MPCP to PCP(1)

Lemma

$\text{MPCP} \leq \text{PCP}$.

Reducibility of MPCP to PCP(1)

Lemma

$\mathrm{MPCP} \leq \mathrm{PCP}$.

Proof.

Let $\#, \$ \notin \Sigma$. For word $w = a_1 a_2 \dots a_m \in \Sigma^+$ define

$$\bar{w} = \#a_1\#a_2\#\dots\#a_m\#$$
$$\grave{w} = \#a_1\#a_2\#\dots\#a_m$$
$$\acute{w} = a_1\#a_2\#\dots\#a_m\#$$

...

Reducibility of MPCP to PCP(1)

Lemma

$\mathrm{MPCP} \leq \mathrm{PCP}$.

Proof.

Let $\#, \$ \notin \Sigma$. For word $w = a_1 a_2 \dots a_m \in \Sigma^+$ define

$$\bar{w} = \#a_1\#a_2\#\dots\#a_m\#$$
$$\grave{w} = \#a_1\#a_2\#\dots\#a_m$$
$$\acute{w} = a_1\#a_2\#\dots\#a_m\#$$

For input $C = ((x_1, y_1), \dots, (x_k, y_k))$ define
$f(C) = ((\bar{x_1}, \grave{y_1}), (\acute{x_1}, \grave{y_1}), (\acute{x_2}, \grave{y_2}), \dots, (\acute{x_k}, \grave{y_k}), (\$, \#\$))$

...

Reducibility of MPCP to PCP(2)

Proof (continued).

$f(C) = ((\bar{x}_1, \grave{y}_1), (\acute{x}_1, \acute{y}_1), (\acute{x}_2, \acute{y}_2), \ldots, (\acute{x}_k, \acute{y}_k), (\$, \#\$))$

Function f is computable, and can suitably get extended to a total function. It holds that
C has a solution with $i_1 = 1$ iff $f(C)$ has a solution:

□

Reducibility of MPCP to PCP(2)

Proof (continued).

$f(C) = ((\bar{x}_1, \grave{y}_1), (\acute{x}_1, \grave{y}_1), (\acute{x}_2, \grave{y}_2), \ldots, (\acute{x}_k, \grave{y}_k), (\$, \#\$))$

Function f is computable, and can suitably get extended to a total function. It holds that
C has a solution with $i_1 = 1$ iff $f(C)$ has a solution:

Let $1, i_2, i_3, \ldots, i_n$ be a solution for C. Then
$1, i_2 + 1, \ldots, i_n + 1, k + 2$ is a solution for $f(C)$.

□

Reducibility of MPCP to PCP(2)

Proof (continued).

$f(C) = ((\bar{x}_1, \grave{y}_1), (\acute{x}_1, \grave{y}_1), (\acute{x}_2, \grave{y}_2), \ldots, (\acute{x}_k, \grave{y}_k), (\$, \#\$))$

Function f is computable, and can suitably get extended to a total function. It holds that
C has a solution with $i_1 = 1$ iff $f(C)$ has a solution:

Let $1, i_2, i_3, \ldots, i_n$ be a solution for C. Then $1, i_2 + 1, \ldots, i_n + 1, k + 2$ is a solution for $f(C)$.

If $i_1, \ldots, i_n$ is a match for $f(C)$, then (due to the construction of the word pairs) there is a $m \leq n$ such that $i_1 = 1, i_m = k + 2$ and $i_j \in \{2, \ldots, k + 1\}$ for $j \in \{2, \ldots, m - 1\}$. Then $1, i_2 - 1, \ldots, i_{m-1} - 1$ is a solution for C.

□

Reducibility of MPCP to PCP(2)

Proof (continued).

$f(C) = ((\bar{x}_1, \grave{y}_1), (\acute{x}_1, \grave{y}_1), (\acute{x}_2, \grave{y}_2), \ldots, (\acute{x}_k, \grave{y}_k), (\$, \#\$))$

Function f is computable, and can suitably get extended to a total function. It holds that
C has a solution with $i_1 = 1$ iff $f(C)$ has a solution:

Let $1, i_2, i_3, \ldots, i_n$ be a solution for C. Then $1, i_2 + 1, \ldots, i_n + 1, k + 2$ is a solution for $f(C)$.

If $i_1, \ldots, i_n$ is a match for $f(C)$, then (due to the construction of the word pairs) there is a $m \leq n$ such that $i_1 = 1, i_m = k + 2$ and $i_j \in \{2, \ldots, k + 1\}$ for $j \in \{2, \ldots, m - 1\}$. Then $1, i_2 - 1, \ldots, i_{m-1} - 1$ is a solution for C.

$\Rightarrow$ f is a reduction from MPCP to PCP. □

PCP: Undecidability – Where are we?

Theorem (Undecidability of PCP)

PCP *is undecidable.*

Proof via an intermediate other problem
modified PCP (MPCP)

1. Reduce MPCP to PCP (MPCP $\leq$ PCP)
2. Reduce halting problem to MPCP ($H \leq$ MPCP)

PCP: Undecidability – Where are we?

Theorem (Undecidability of PCP)

PCP *is undecidable.*

Proof via an intermediate other problem
modified PCP (MPCP)

1. Reduce MPCP to PCP (MPCP $\leq$ PCP) ✓
2. Reduce halting problem to MPCP ($H \leq$ MPCP)

PCP: Undecidability – Where are we?

Theorem (Undecidability of PCP)

PCP *is undecidable.*

Proof via an intermediate other problem
modified PCP (MPCP)

1. Reduce MPCP to PCP (MPCP $\leq$ PCP) ✓
2. Reduce halting problem to MPCP ($H \leq$ MPCP)

Reducibility of H to MPCP(1)

Lemma

$H \leq \mathrm{MPCP}$.

Proof.

Goal: Construct for Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, E)$ and word $w \in \Sigma^*$ an MPCP instance $C = ((x_1, y_1), \ldots, (x_k, y_k))$ such that

$$M \text{ started on } w \text{ terminates iff } C \in \mathrm{MPCP}.$$

...

Reducibility of H to MPCP(2)

Proof (continued).

Idea:

- Sequence of words describes sequence of configurations of the TM
- "x-row" follows "y-row"

x : # c_0 # c_1 # c_2 #

y : # c_0 # c_1 # c_2 # c_3 #

- Configurations get mostly just copied, only the area around the head changes.
- After a terminating configuration has been reached: make row equal by deleting the configuration.

...

Reducibility of H to MPCP(3)

Proof (continued).

Alphabet of C is $\Gamma \cup Q \cup \{\#\}$.

1. Pair: $(\#, \#\square q_0 w\#)$

Other pairs:

1. copy: (a, a) for all $a \in \Gamma \cup \{\#\}$
2. transition:

$$
\begin{aligned}
(qa, q'c) &\text{ if } \delta(q,a) = (q', c, N)\\
(qa, cq') &\text{ if } \delta(q,a) = (q', c, R)\\
(bqa, q'bc) &\text{ if } \delta(q,a) = (q', c, L) \text{ for all } b \in \Gamma\\
(\#qa, \#q'\square c) &\text{ if } \delta(q,a) = (q', c, L)
\end{aligned}
$$

...

Reducibility of H to MPCP(4)

Proof (continued).

$$(q\#, q'c\#) \text{ if } \delta(q, \square) = (q', c, N)$$
$$(q\#, cq'\#) \text{ if } \delta(q, \square) = (q', c, R)$$
$$(bq\#, q'bc\#) \text{ if } \delta(q, \square) = (q', c, L) \text{ for all } b \in \Gamma$$

3. deletion: (aq_e, q_e) and $(q_e a, q_e)$ for all $a \in \Gamma$ and $q_e \in E$
4. finish: $(q_e\#\#, \#)$ for all $q_e \in E$

...

Reducibility of H to MPCP(5)

Proof (continued).

"$\Rightarrow$" If M terminates on input w, there is a sequence of $c_0, \ldots, c_t$ of configurations with

- $c_0 = \square q_0 w$ is the start configuration
- c_t is an end configuration
  ($c_t = u q_e v$ mit $u, v \in \Gamma^*$ and $q_e \in E$)
- $c_i \vdash c_{i+1}$ for $i = 0, 1, \ldots, t-1$

...

Reducibility of H to MPCP(5)

Proof (continued).

"$\Rightarrow$" If M terminates on input w, there is a sequence of $c_0, \ldots, c_t$ of configurations with

- $c_0 = \square q_0 w$ is the start configuration
- c_t is an end configuration
  ($c_t = u q_e v$ mit $u, v \in \Gamma^*$ and $q_e \in E$)
- $c_i \vdash c_{i+1}$ for $i = 0, 1, \ldots, t-1$

Then C has a match with the overall word

$$\#c_0\#c_1\#\ldots\#c_t\#c_t'\#c_t''\#\ldots\#q_e\#\#$$

Up to c_t: "'x-row"' follows "'y-row"'

From c_t': deletion of symbols adjacent to q_e. ...

Reducibility of H to MPCP(6)

Proof (continued).

"$\Leftarrow$" If C has a solution, it has the form

$$\#c_0\#c_1\#\ldots\#c_n\#\#,$$

with $c_0 = \square q_0 w$. Moreover, there is an $\ell \leq n$, such that an end state q_e occurs for the first time in c_ℓ.
All c_i for $i \leq \ell$ are configurations of M and $c_i \vdash c_{i+1}$ for $i \in \{0, \ldots, \ell - 1\}$.
$c_0, \ldots, c_\ell$ is hence the sequence of configurations of M on input w, which shows that the TM terminates. $\square$

PCP: Undecidability – Done!

Theorem (Undecidability of PCP)

PCP *is undecidable.*

Proof via an intermediate other problem
modified PCP (MPCP)

1. Reduce MPCP to PCP (MPCP $\leq$ PCP) ✓
2. Reduce halting problem to MPCP ($H \leq$ MPCP)

PCP: Undecidability – Done!

Theorem (Undecidability of PCP)

PCP *is undecidable*.

Proof via an intermediate other problem
modified PCP (MPCP)

1. Reduce MPCP to PCP (MPCP $\leq$ PCP) ✓
2. Reduce halting problem to MPCP ($H \leq$ MPCP) ✓

PCP: Undecidability – Done!

Theorem (Undecidability of PCP)

PCP *is undecidable.*

Proof via an intermediate other problem
modified PCP (MPCP)

1. Reduce MPCP to PCP (MPCP $\leq$ PCP) ✓
2. Reduce halting problem to MPCP ($H \leq$ MPCP) ✓

Proof.

Due to $H \leq$ MPCP and MPCP $\leq$ PCP it holds that $H \leq$ PCP.
Since H is undecidable, also PCP must be undecidable. □

PCP with $\Sigma = \{0, 1\}$

Theorem

The Post correspondence problem is already undecidable if the alphabet is restricted to $\{0, 1\}$*.*

Proof by reduction from the general PCP.

Further Undecidable Problems

And What Else?

- Here we conclude our discussion of undecidable problems.
- Many more undecidable problems exist.
- In this section, we briefly discuss some further classical results.

Undecidable Grammar Problems

Some Grammar Problems

Given context-free grammars G_1 and G_2, ...

- ... is $\mathcal{L}(G_1) \cap \mathcal{L}(G_2) = \emptyset$?
- ... is $|\mathcal{L}(G_1) \cap \mathcal{L}(G_2)| = \infty$?
- ... is $\mathcal{L}(G_1) \cap \mathcal{L}(G_2)$ context-free?
- ... is $\mathcal{L}(G_1) \subseteq \mathcal{L}(G_2)$?
- ... is $\mathcal{L}(G_1) = \mathcal{L}(G_2)$?

Given a context-sensitive grammar G, ...

- ... is $\mathcal{L}(G) = \emptyset$?
- ... is $|\mathcal{L}(G)| = \infty$?

$\rightsquigarrow$ all undecidable by reduction from PCP
(see Schöning, Chapter 2.8)

Gödel's First Incompleteness Theorem (1)

Definition (Arithmetic Formula)

An arithmetic formula is a closed predicate logic formula using

- constant symbols 0 and 1,
- function symbols $+$ and $\cdot$, and
- equality ($=$) as the only relation symbols.

It is called true if it is true under the usual interpretation of 0, 1, $+$ and $\cdot$ over $\mathbb{N}_0$.

German: arithmetische Formel

Beispiel: $\forall x \exists y \forall z(((x \cdot y) = z) \wedge ((1 + x) = (x \cdot y)))$

Gödel's First Incompleteness Theorem (2)

Gödel's First Incompleteness Theorem

The problem of deciding if a given arithmetic formula is true is undecidable.

Moreover, neither it nor its complement are semi-decidable.

As a consequence, there exists no sound and complete proof system for arithmetic formulas.

German: erster Gödelscher Unvollständigkeitssatz

Summary

Summary

- Post Correspondence Problem:
  Find a sequence of word pairs s.t. the concatenation of all first components equals the one of all second components.
- The Post Correspondence Problem is semi-decidable but not decidable.

What's Next?

contents of this course:

A. background ✓
   - ▷ mathematical foundations and proof techniques

B. logic ✓
   - ▷ How can knowledge be represented?
     How can reasoning be automated?

C. automata theory and formal languages ✓
   - ▷ What is a computation?

D. Turing computability
   - ▷ What can be computed at all?

E. complexity theory
   - ▷ What can be computed efficiently?

F. more computability theory
   - ▷ Other models of computability

What's Next?

contents of this course:

A. background ✓
   - ▷ mathematical foundations and proof techniques

B. logic ✓
   - ▷ How can knowledge be represented?
     How can reasoning be automated?

C. automata theory and formal languages ✓
   - ▷ What is a computation?

D. Turing computability ✓
   - ▷ What can be computed at all?

E. complexity theory
   - ▷ What can be computed efficiently?

F. more computability theory
   - ▷ Other models of computability