Computability and Complexity

Lecture 6

Reductions via Computation Histories
Undecidability of Emptiness of Linear Bounded Automata
Undecidability of Post Correspondence Problem

given by Jiri Srba

Reduction via Computation Histories

Recall Reduction from A to B

- A language A is reducible to language B iff a decider for B can be used to algorithmically construct a decider for language A.

If A is reducible to B and A is undecidable, then B is undecidable.

Reduction via Computation Histories

Recall Reduction from A to B

- A language A is reducible to language B iff a decider for B can be used to algorithmically construct a decider for language A.

If A is reducible to B and A is undecidable, then B is undecidable.

Definition (Accepting/Rejecting Computation History)

Let M be a TM. A computation history of M on input w is a sequence of configurations $C_1, C_2, \ldots, C_\ell$ such that:

- $C_1 = q_0 w$ is the initial configuration,
- C_i yields C_{i+1} for all $1 \leq i < \ell$, and
- C_ℓ is a halting configuration (in either accept or reject state).

If C_ℓ is accepting, then the history is called accepting.
If C_ℓ is rejecting, then the history is called rejecting.

M accepts w iff M on w has an accepting computation history.

Linear Bounded Automaton and the Emptiness Problem

Definition

Linear bounded automaton (LBA) is a restricted Turing machine M such that when M runs on any input string w, its head always stays within the first $|w|$ cells.

Theorem

The language A_{LBA} is decidable.

Linear Bounded Automaton and the Emptiness Problem

Definition

Linear bounded automaton (LBA) is a restricted Turing machine M such that when M runs on any input string w, its head always stays within the first $|w|$ cells.

Theorem

The language A_{LBA} is decidable.

Emptiness Problem: "Given an LBA B, is $L(B) = \emptyset$?"

$$E_{LBA} \stackrel{\text{def}}{=} \{\langle B\rangle \mid \ B \text{ is an LBA such that } L(B) = \emptyset \ \}$$

Theorem

The language E_{LBA} is undecidable.

Proof: By reduction from A_{TM} to E_{LBA} via computation histories.

Proof (Reduction for A_{TM} to E_{LBA})

1. Assume that we have a decider R for E_{LBA}.
2. Using R, we construct a decider S for A_{TM}:

   $S =$ " On input $\langle M, w\rangle$:
   1. From M and w build an LBA B such that
      $L(B) \neq \emptyset$ if and only if M accepts w
   2. Run R (decider for E_{LBA}) on $\langle B\rangle$.
   3. If R accepted then $\underline{S \text{ rejects}}$.
      If R rejected then $\underline{S \text{ accepts}}$. "
3. We know that S cannot exist, and hence R cannot exist either.
4. Conclusion: E_{LBA} is undecidable.

Proof (Reduction for A_{TM} to E_{LBA})

1. Assume that we have a decider R for E_{LBA}.
2. Using R, we construct a decider S for A_{TM}:

   $S =$ " On input $\langle M, w\rangle$:
   1. From M and w build an LBA B such that
      $L(B) \neq \emptyset$ if and only if M accepts w
   2. Run R (decider for E_{LBA}) on $\langle B\rangle$.
   3. If R accepted then <u>S rejects</u>.
      If R rejected then <u>S accepts</u>. "
3. We know that S cannot exist, and hence R cannot exist either.
4. Conclusion: E_{LBA} is undecidable.

TO DO (The Tricky Part)

From M and w construct an LBA B s.t. $L(B) \neq \emptyset$ iff M accepts w.

Proof (Construction of LBA B from M and w)

Idea:

- We construct B such that it accepts exactly all strings of the form

$$\#C_1\#C_2\#C_3\#\ldots\#C_\ell\#$$

  where $C_1, C_2, C_3, \ldots, C_\ell$ is an accepting computation history of M on w.

- Now clearly $L(B) \neq \emptyset$ if and only if M accepts w.

Proof (Construction of LBA B from M and w)

$B =$ " On input x:

1. If x is not of the form $\#C_1\#C_2\#\ldots\#C_\ell\#$ for some strings $C_1, \ldots, C_\ell$ then $\underline{B \text{ rejects}}$.
2. Verify whether $\#C_1\#C_2\#\ldots\#C_\ell\#$ satisfies the following three conditions:
   a) $C_1 = q_0 w$
   b) C_ℓ is an accept configuration
   c) C_i yields C_{i+1} for all i (zigzag between them)
3. If all three conditions are true, then $\underline{S \text{ accepts}}$, else $\underline{S \text{ rejects}}$. "

Proof (Construction of LBA B from M and w)

$B =$ " On input x:

1. If x is not of the form $\#C_1\#C_2\#\ldots\#C_\ell\#$ for some strings $C_1, \ldots, C_\ell$ then $\underline{B \text{ rejects}}$.
2. Verify whether $\#C_1\#C_2\#\ldots\#C_\ell\#$ satisfies the following three conditions:
   a) $C_1 = q_0 w$
   b) C_ℓ is an accept configuration
   c) C_i yields C_{i+1} for all i (zigzag between them)
3. If all three conditions are true, then $\underline{S \text{ accepts}}$, else $\underline{S \text{ rejects}}$. "

Notice

- The constructed machine B is LBA.
- We actually never run B, it is merely the input for R (the decider for E_{LBA}) in order to achieve a contradiction.

More Undecidable Problems from Language Theory

Problem: "Given a CFG G, is $L(G) = \Sigma^*$?"

$$ALL_{CFG} \stackrel{\text{def}}{=} \{\langle G \rangle \mid \ G \text{ is a CFG such that } L(G) = \Sigma^* \ \}$$

Theorem

The language ALL_{CFG} is undecidable.

Proof: very interesting technique based on computation histories (optional reading in the book). □

$$EQ_{CFG} \stackrel{\text{def}}{=} \{\langle G_1, G_2 \rangle \mid \ G_1 \text{ and } G_2 \text{ are CFGs s.t. } L(G_1) = L(G_2) \ \}$$

Theorem

The language EQ_{CFG} is undecidable.

Proof: By reduction from ALL_{CFG}. Next tutorial. □

Post Correspondence Problem (Emil Post, 1946)

Instance of the Post Correspondence Problem (PCP):

A PCP instance over Σ is a finite collection P of dominos

$$P = \{ \ [\frac{t_1}{b_1}], [\frac{t_2}{b_2}], \cdots, [\frac{t_k}{b_k}] \ \}$$

where for all i, $1 \leq i \leq k$, we have $t_i, b_i \in \Sigma^+$.

Post Correspondence Problem (Emil Post, 1946)

Instance of the Post Correspondence Problem (PCP):

A PCP instance over Σ is a finite collection P of dominos

$$P = \{ \ [\frac{t_1}{b_1}], [\frac{t_2}{b_2}], \cdots, [\frac{t_k}{b_k}] \ \}$$

where for all i, $1 \leq i \leq k$, we have $t_i, b_i \in \Sigma^+$.

Match:

Assume a given PCP instance P. A match is a nonempty sequence

$$i_1, i_2, \ldots, i_\ell$$

of numbers from $\{1, 2, \ldots, k\}$ (repeating is allowed) such that

$$t_{i_1} t_{i_2} \ldots t_{i_\ell} = b_{i_1} b_{i_2} \ldots b_{i_\ell} \ .$$

Post Correspondence Problem (PCP)

Question:

Does a given PCP instance P have a match?

Language formulation:

$$PCP \stackrel{\text{def}}{=} \{\langle P\rangle \mid \ P \text{ is a PCP instance and it has a match } \}$$

Post Correspondence Problem (PCP)

Question:

Does a given PCP instance P have a match?

Language formulation:

$$PCP \stackrel{\text{def}}{=} \{\langle P\rangle \mid \ P \text{ is a PCP instance and it has a match } \}$$

Theorem

The language PCP is undecidable.

Proof: By reduction via computation histories from A_{TM}.

Proof Structure (Undecidability of PCP)

The reduction will work in two steps:

1. We reduce A_{TM} to $MPCP$.
2. We reduce $MPCP$ to PCP.

MPCP (Modified PCP):

$$MPCP \stackrel{\text{def}}{=} \{\langle P\rangle \mid \text{ } P \text{ is a PCP instance and it has a match which starts with index 1 } \}$$

In reduction from A_{TM} we will without loss of generality assume that on input $\langle M, w\rangle$ of A_{TM} the machine M never attempts to move its head off the left-hand end of the tape.

Proof (Reduction from A_{TM} to $MPCP$)

For input $\langle M, w \rangle$ of A_{TM} construct a $MPCP$ instance P such that

M accepts w iff P has a match starting with domino 1.

1. Add a start (first) domino $\left[\dfrac{\#}{\#q_0 w\#}\right]$.

Proof (Reduction from A_{TM} to $MPCP$)

For input $\langle M, w \rangle$ of A_{TM} construct a $MPCP$ instance P such that

M accepts w iff P has a match starting with domino 1.

1. Add a start (first) domino $\left[\dfrac{\#}{\#q_0w\#}\right]$.
2. If $\delta(q, a) = (r, b, R)$ add the domino $\left[\dfrac{qa}{br}\right]$.

Proof (Reduction from A_{TM} to $MPCP$)

For input $\langle M, w \rangle$ of A_{TM} construct a $MPCP$ instance P such that

M accepts w iff P has a match starting with domino 1.

1. Add a start (first) domino $\left[\frac{\#}{\#q_0w\#}\right]$.
2. If $\delta(q, a) = (r, b, R)$ add the domino $\left[\frac{qa}{br}\right]$.
3. If $\delta(q, a) = (r, b, L)$ add the domino $\left[\frac{cqa}{rcb}\right]$ for all $c \in \Gamma$.

Proof (Reduction from A_{TM} to $MPCP$)

For input $\langle M, w\rangle$ of A_{TM} construct a $MPCP$ instance P such that

M accepts w iff P has a match starting with domino 1.

1. Add a start (first) domino $\left[\frac{\#}{\#q_0w\#}\right]$.
2. If $\delta(q, a) = (r, b, R)$ add the domino $\left[\frac{qa}{br}\right]$.
3. If $\delta(q, a) = (r, b, L)$ add the domino $\left[\frac{cqa}{rcb}\right]$ for all $c \in \Gamma$.
4. Add the domino $\left[\frac{a}{a}\right]$ for all $a \in \Gamma$.

Proof (Reduction from A_{TM} to $MPCP$)

For input $\langle M, w\rangle$ of A_{TM} construct a $MPCP$ instance P such that

M accepts w iff P has a match starting with domino 1.

1. Add a start (first) domino $\left[\frac{\#}{\# q_0 w \#}\right]$.
2. If $\delta(q, a) = (r, b, R)$ add the domino $\left[\frac{qa}{br}\right]$.
3. If $\delta(q, a) = (r, b, L)$ add the domino $\left[\frac{cqa}{rcb}\right]$ for all $c \in \Gamma$.
4. Add the domino $\left[\frac{a}{a}\right]$ for all $a \in \Gamma$.
5. Add the dominos $\left[\frac{\#}{\#}\right]$ and $\left[\frac{\#}{\sqcup\#}\right]$.

Proof (Reduction from A_{TM} to $MPCP$)

For input $\langle M, w \rangle$ of A_{TM} construct a $MPCP$ instance P such that

M accepts w iff P has a match starting with domino 1.

1. Add a start (first) domino $\left[\frac{\#}{\#q_0 w\#}\right]$.
2. If $\delta(q, a) = (r, b, R)$ add the domino $\left[\frac{qa}{br}\right]$.
3. If $\delta(q, a) = (r, b, L)$ add the domino $\left[\frac{cqa}{rcb}\right]$ for all $c \in \Gamma$.
4. Add the domino $\left[\frac{a}{a}\right]$ for all $a \in \Gamma$.
5. Add the dominos $\left[\frac{\#}{\#}\right]$ and $\left[\frac{\#}{\sqcup\#}\right]$.
6. Add the dominos $\left[\frac{aq_{accept}}{q_{accept}}\right]$ and $\left[\frac{q_{accept}a}{q_{accept}}\right]$ for all $a \in \Gamma$.

Proof (Reduction from A_{TM} to $MPCP$)

For input $\langle M, w\rangle$ of A_{TM} construct a $MPCP$ instance P such that

M accepts w iff P has a match starting with domino 1.

1. Add a start (first) domino $\left[\frac{\#}{\#q_0w\#}\right]$.
2. If $\delta(q,a) = (r,b,R)$ add the domino $\left[\frac{qa}{br}\right]$.
3. If $\delta(q,a) = (r,b,L)$ add the domino $\left[\frac{cqa}{rcb}\right]$ for all $c \in \Gamma$.
4. Add the domino $\left[\frac{a}{a}\right]$ for all $a \in \Gamma$.
5. Add the dominos $\left[\frac{\#}{\#}\right]$ and $\left[\frac{\#}{\sqcup\#}\right]$.
6. Add the dominos $\left[\frac{aq_{accept}}{q_{accept}}\right]$ and $\left[\frac{q_{accept}a}{q_{accept}}\right]$ for all $a \in \Gamma$.
7. Finally add the domino $\left[\frac{q_{accept}\#\#}{\#}\right]$.

Proof (Reduction from *MPCP* to *PCP*)

Conclusion

MPCP is undecidable.

Proof (Reduction from *MPCP* to *PCP*)

Conclusion

MPCP is undecidable.

Now we want to reduce *MPCP* to *PCP*:

Given an instance P of MPCP we build an instance P' of PCP s.t.

P has a match starting with domino 1 iff P' has a match.

Let $w = a_1 a_2 \dots a_n$ be a string. We use the notation

- $*w \stackrel{\text{def}}{=} *a_1 * a_2 * \dots * a_n$,
- $w* \stackrel{\text{def}}{=} a_1 * a_2 * \dots * a_n*$, and
- $*w* \stackrel{\text{def}}{=} *a_1 * a_2 * \dots * a_n*$.

Proof (Reduction from *MPCP* to *PCP*)

Given an instance P of MPCP we build an instance P' of PCP s.t.

P has a match starting with domino 1 iff P' has a match.

Construction of P' from P (here $*$ and $\diamond$ are fresh symbols):

Proof (Reduction from *MPCP* to *PCP*)

Given an instance P of MPCP we build an instance P' of PCP s.t.

P has a match starting with domino 1 iff P' has a match.

Construction of P' from P (here $*$ and $\diamond$ are fresh symbols):

- For the first domino $\left[\frac{t_1}{b_1}\right]$ in P we add $\left[\frac{*t_1}{*b_1*}\right]$ to P'.

Proof (Reduction from *MPCP* to *PCP*)

Given an instance P of MPCP we build an instance P' of PCP s.t.

P has a match starting with domino 1 iff P' has a match.

Construction of P' from P (here $*$ and $\diamond$ are fresh symbols):

- For the first domino $\left[\frac{t_1}{b_1}\right]$ in P we add $\left[\frac{*t_1}{*b_1*}\right]$ to P'.
- For all dominos $\left[\frac{t_i}{b_i}\right]$ in P we add the dominos $\left[\frac{*t_i}{b_i*}\right]$ to P'.

Proof (Reduction from *MPCP* to *PCP*)

Given an instance P of MPCP we build an instance P' of PCP s.t.

P has a match starting with domino 1 iff P' has a match.

Construction of P' from P (here $*$ and $\diamond$ are fresh symbols):

- For the first domino $\left[\frac{t_1}{b_1}\right]$ in P we add $\left[\frac{*t_1}{*b_1*}\right]$ to P'.
- For all dominos $\left[\frac{t_i}{b_i}\right]$ in P we add the dominos $\left[\frac{*t_i}{b_i*}\right]$ to P'.
- We add the domino $\left[\frac{*\diamond}{\diamond}\right]$ to P'.

Proof (Reduction from *MPCP* to *PCP*)

Given an instance P of MPCP we build an instance P' of PCP s.t.

P has a match starting with domino 1 iff P' has a match.

Construction of P' from P (here $*$ and $\diamond$ are fresh symbols):

- For the first domino $\left[\frac{t_1}{b_1}\right]$ in P we add $\left[\frac{*t_1}{*b_1*}\right]$ to P'.
- For all dominos $\left[\frac{t_i}{b_i}\right]$ in P we add the dominos $\left[\frac{*t_i}{b_i*}\right]$ to P'.
- We add the domino $\left[\frac{*\diamond}{\diamond}\right]$ to P'.

It is easy to see that if in P (where $i_1 = 1$)

$$t_{i_1} t_{i_2} \ldots t_{i_\ell} = b_{i_1} b_{i_2} \ldots b_{t_\ell}$$

then in P'

$$*t_{i_1} * t_{i_2} * \ldots * t_{i_\ell} * \diamond = *b_{i_1} * b_{i_2} * \ldots * b_{t_\ell} * \diamond$$

and vice verse.

Conclusion

PCP is undecidable.

Facts:

Undecidability of *PCP* can be further used to show that e.g. the following problems are undecidable too:

- "Is a given CFG ambiguous?"
- "Given CFGs G_1 and G_2 is $L(G_1) \cap L(G_2) = \emptyset$?"
- And many more ...

Exam Questions

- Undecidability of emptiness for LBA.
- PCP and MPCP definitions and examples.
- Undecidability proofs of MPCP and PCP (two reductions).