CS310 : Automata Theory 2019

Lecture 28: Reductions

Instructor: S. Akshay

IITB, India

14-03-2019

Recap of previous lecture

Regular $\subsetneq$ Decidable $\subsetneq$ Recursively Enumerable $\subsetneq$ All languages

DFA/NFA $<$ Algorithms/Halting TM $<$ Semi-algorithms/TM

Properties

1. There exist languages that are not R.E.
2. There exist languages that are R.E but are undecidable.
   Eg. universal TM lang $L^A_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
3. Decidable languages are closed under complementation.
4. L is decidable iff L is $R.E$ and $\overline{L}$ is also R.E.

The halting problem

The halting problem for Turing Machines is undecidable

Does a given Turing machine halt on a given input?

- $L_{TM}^{HALT} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ halts on input } w\}$.

The halting problem

The halting problem for Turing Machines is undecidable

Does a given Turing machine halt on a given input?

- $L_{TM}^{HALT} = \{\langle M, w\rangle \mid M$ is a TM and M halts on input $w\}$.

Proof: Suppose there exists TM H deciding L_{TM}^{HALT}, then construct a TM D s.t., on input $\langle M, w\rangle$:

- runs TM H on input $\langle M, w\rangle$
- if H rejects then reject.
- if H accepts, then simulate M on w until it halts.
- if at halting M has accepted w, accept, else reject.

But D decides L_{TM}^{A} which is undecidable. A contradiction.

The halting problem

The halting problem for Turing Machines is undecidable

Does a given Turing machine halt on a given input?

- $L_{TM}^{HALT} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ halts on input } w\}$.

Proof: Suppose there exists TM H deciding L_{TM}^{HALT}, then construct a TM D s.t., on input $\langle M, w\rangle$:

- runs TM H on input $\langle M, w\rangle$
- if H rejects then reject.
- if H accepts, then simulate M on w until it halts.
- if at halting M has accepted w, accept, else reject.

But D decides L_{TM}^{A} which is undecidable. A contradiction.

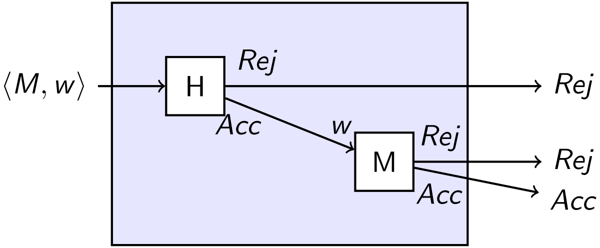
This proof strategy is called a reduction.

Reduction from the acceptance problem

The halting problem for Turing Machines is undecidable

Does a given Turing machine halt on a given input?

- $L_{TM}^{HALT} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ halts on input } w\}$.



Some more undecidable problems

The emptiness problem for TMs

Does a given Turing machine accept any word?

- $L_{TM}^{\emptyset} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$.

Some more undecidable problems

The emptiness problem for TMs is undecidable

Does a given Turing machine accept any word?

- $L^{\emptyset}_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$.

Proof:

- For contradiction, assume its decidable, say by TM R.

Some more undecidable problems

The emptiness problem for TMs is undecidable

Does a given Turing machine accept any word?

- $L^{\emptyset}_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$.

Proof:

- For contradiction, assume its decidable, say by TM R.
- Define M_1 which on input x,
  - if $x \neq w$, it rejects
  - if $x = w$, run M on w and accept if M accepts.

Some more undecidable problems

The emptiness problem for TMs is undecidable

Does a given Turing machine accept any word?

- $L^{\emptyset}_{TM} = \{\langle M\rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$.

Proof:

- For contradiction, assume its decidable, say by TM R.
- Define M_1 which on input x,
  - if $x \neq w$, it rejects
  - if $x = w$, run M on w and accept if M accepts.
- Construct S which decides A_{TM}: Given input $\langle M, w\rangle$,

Some more undecidable problems

The emptiness problem for TMs is undecidable

Does a given Turing machine accept any word?

- $L^{\emptyset}_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$.

Proof:

- For contradiction, assume its decidable, say by TM R.
- Define M_1 which on input x,
  - if $x \neq w$, it rejects
  - if $x = w$, run M on w and accept if M accepts.
- Construct S which decides A_{TM}: Given input $\langle M, w \rangle$,
  - Construct M_1 on tape from M and w,

Some more undecidable problems

The emptiness problem for TMs is undecidable

Does a given Turing machine accept any word?

- $L^{\emptyset}_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$.

Proof:

- For contradiction, assume its decidable, say by TM R.
- Define M_1 which on input x,
  - if $x \neq w$, it rejects
  - if $x = w$, run M on w and accept if M accepts.
- Construct S which decides A_{TM}: Given input $\langle M, w \rangle$,
  - Construct M_1 on tape from M and w,
  - Run R on input $\langle M_1 \rangle$.

Some more undecidable problems

The emptiness problem for TMs is undecidable

Does a given Turing machine accept any word?

- $L^{\emptyset}_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$.

Proof:

- For contradiction, assume its decidable, say by TM R.
- Define M_1 which on input x,
  - if $x \neq w$, it rejects
  - if $x = w$, run M on w and accept if M accepts.
- Construct S which decides A_{TM}: Given input $\langle M, w \rangle$,
  - Construct M_1 on tape from M and w,
  - Run R on input $\langle M_1 \rangle$.
  - If R accepts,

Some more undecidable problems

The emptiness problem for TMs is undecidable

Does a given Turing machine accept any word?

- $L^{\emptyset}_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$.

Proof:

- For contradiction, assume its decidable, say by TM R.
- Define M_1 which on input x,
  - if $x \neq w$, it rejects
  - if $x = w$, run M on w and accept if M accepts.
- Construct S which decides A_{TM}: Given input $\langle M, w \rangle$,
  - Construct M_1 on tape from M and w,
  - Run R on input $\langle M_1 \rangle$.
  - If R accepts, *reject*; if R rejects then *accept*.

Some more undecidable problems

The regularity problem for TMs

Does a given Turing machine accept a regular language?

- $L_{TM}^{REG} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is a regular language } \}$.

Some more undecidable problems

The regularity problem for TMs

Does a given Turing machine accept a regular language?

- $L_{TM}^{REG} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is a regular language } \}$.

Proof: By contradiction, assume R decides Reg. We define S

Some more undecidable problems

The regularity problem for TMs

Does a given Turing machine accept a regular language?

- $L_{TM}^{REG} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is a regular language } \}$.

Proof: By contradiction, assume R decides Reg. We define S

- $S =$ On input $\langle M, w \rangle$;
  - construct TM M_2 as follows:

Some more undecidable problems

The regularity problem for TMs

Does a given Turing machine accept a regular language?

- $L_{TM}^{REG} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is a regular language } \}$.

Proof: By contradiction, assume R decides Reg. We define S

- $S =$ On input $\langle M, w \rangle$;
  - construct TM M_2 as follows:
    M_2 on input x does the foll:
    - if x has the form 0^n1^n, accept.
    - If not, run M on w and accept if M accepts w.

Some more undecidable problems

The regularity problem for TMs

Does a given Turing machine accept a regular language?

- $L_{TM}^{REG} = \{\langle M \rangle \mid M$ is a TM and $L(M)$ is a regular language $\}$.

Proof: By contradiction, assume R decides Reg. We define S

- $S =$ On input $\langle M, w \rangle$;
  - construct TM M_2 as follows:
    M_2 on input x does the foll:
    - if x has the form $0^n 1^n$, accept.
    - If not, run M on w and accept if M accepts w.
  - Run R on $\langle M_2 \rangle$.

Some more undecidable problems

The regularity problem for TMs

Does a given Turing machine accept a regular language?

- $L_{TM}^{REG} = \{\langle M \rangle \mid M$ is a TM and $L(M)$ is a regular language $\}$.

Proof: By contradiction, assume R decides Reg. We define S

- $S =$ On input $\langle M, w \rangle$;
  - construct TM M_2 as follows:
    M_2 on input x does the foll:
    - if x has the form $0^n 1^n$, accept.
    - If not, run M on w and accept if M accepts w.
  - Run R on $\langle M_2 \rangle$.
  - If R accepts, *accept*; if R rejects, *reject*.

The principle of reduction

Reduction: An algorithm (TM!) to convert instances of one problem to another

- Convert instance of P_1 to instance of P_2
- answer is yes for P_1 iff answer is yes for P_2

The principle of reduction

Reduction: An algorithm (TM!) to convert instances of one problem to another

- Convert instance of P_1 to instance of P_2
- answer is yes for P_1 iff answer is yes for P_2
- then P_2 is at least as hard as P_1, i.e.,

The principle of reduction

Reduction: An algorithm (TM!) to convert instances of one problem to another

- Convert instance of P_1 to instance of P_2
- answer is yes for P_1 iff answer is yes for P_2
- then P_2 is at least as hard as P_1, i.e.,
  - if P_1 is not decidable, neither is P_2

The principle of reduction

Reduction: An algorithm (TM!) to convert instances of one problem to another

- Convert instance of P_1 to instance of P_2
- answer is yes for P_1 iff answer is yes for P_2
- then P_2 is at least as hard as P_1, i.e.,
  - if P_1 is not decidable, neither is P_2
  - if P_1 is not r.e., neither is P_2.

The principle of reduction

Reduction: An algorithm (TM!) to convert instances of one problem to another

- Convert instance of P_1 to instance of P_2
- answer is yes for P_1 iff answer is yes for P_2
- then P_2 is at least as hard as P_1, i.e.,
  - if P_1 is not decidable, neither is P_2
  - if P_1 is not r.e., neither is P_2.
- note that every instance of P_2 need not be covered!

Some more undecidable problems

The regularity problem for TMs

Does a given Turing machine accept a regular language?

- $L^{EQ}_{TM} = \{\langle M_1, M_2 \rangle \mid M$ are TMs and $L(M_1) = L(M_2)\}$.

Some more undecidable problems

The regularity problem for TMs

Does a given Turing machine accept a regular language?

- $L_{TM}^{EQ} = \{\langle M_1, M_2 \rangle \mid M$ are TMs and $L(M_1) = L(M_2)\}$.

Proof: Exercise.