CS310 : Automata Theory 2019

Lecture 29: Reductions contd.

Instructor: S. Akshay

IITB, India

18-03-2019

Pop-Quiz

Which is easier: Emptiness or non-emptiness?

- $L_e = \{\langle M \rangle \mid L(M) = \emptyset\}$
- $L_{ne} = \{\langle M \rangle \mid L(M) \neq \emptyset\}$

Recap

Turing machines and computability

1. Definition of Turing machines: high level and low-level descriptions
2. Variants of Turing machines
3. Decidable and Turing recognizable languages

Recap

Turing machines and computability

1. Definition of Turing machines: high level and low-level descriptions
2. Variants of Turing machines
3. Decidable and Turing recognizable languages
4. Church-Turing Hypothesis
5. Undecidability and a proof technique by diagonalization
   - A universal TM lang $L^A_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$

Recap

Turing machines and computability

1. Definition of Turing machines: high level and low-level descriptions
2. Variants of Turing machines
3. Decidable and Turing recognizable languages
4. Church-Turing Hypothesis
5. Undecidability and a proof technique by diagonalization
   - A universal TM lang $L_{TM}^{A} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$

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

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

Recap

Turing machines and computability

1. Definition of Turing machines: high level and low-level descriptions
2. Variants of Turing machines
3. Decidable and Turing recognizable languages
4. Church-Turing Hypothesis
5. Undecidability and a proof technique by diagonalization
   - A universal TM lang $L^A_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
6. Reductions.

Recap

Turing machines and computability

1. Definition of Turing machines: high level and low-level descriptions
2. Variants of Turing machines
3. Decidable and Turing recognizable languages
4. Church-Turing Hypothesis
5. Undecidability and a proof technique by diagonalization
   - A universal TM lang $L^A_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
6. Reductions.
7. Today: Reductions and moar undecidability!

The principle of reduction

(Many-to-one) Reduction

- An algorithm (halting TM!) to convert instances of a problem P_1 to another P_2 such that,
  - answer is yes for P_1 iff answer is yes for P_2
  - answer is no for P_1 iff answer is no for P_2

Note that every instance of P_2 need not be covered!

The principle of reduction

(Many-to-one) Reduction

- An algorithm (halting TM!) to convert instances of a problem P_1 to another P_2 such that,
  - answer is yes for P_1 iff answer is yes for P_2
  - answer is no for P_1 iff answer is no for P_2

Note that every instance of P_2 need not be covered!

Theorem: If there is a reduction from P_1 to P_2,

(then P_2 is at least as hard as P_1, i.e.,)

The principle of reduction

(Many-to-one) Reduction

- An algorithm (halting TM!) to convert instances of a problem P_1 to another P_2 such that,
  - answer is yes for P_1 iff answer is yes for P_2
  - answer is no for P_1 iff answer is no for P_2

Note that every instance of P_2 need not be covered!

Theorem: If there is a reduction from P_1 to P_2,

(then P_2 is at least as hard as P_1, i.e.,)

- if P_1 is undecidable, then so is P_2

The principle of reduction

(Many-to-one) Reduction

- An algorithm (halting TM!) to convert instances of a problem P_1 to another P_2 such that,
  - answer is yes for P_1 iff answer is yes for P_2
  - answer is no for P_1 iff answer is no for P_2

Note that every instance of P_2 need not be covered!

Theorem: If there is a reduction from P_1 to P_2,

(then P_2 is at least as hard as P_1, i.e.,)

- if P_1 is undecidable, then so is P_2
- if P_1 is not r.e., then so is P_2.

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?

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

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.

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.

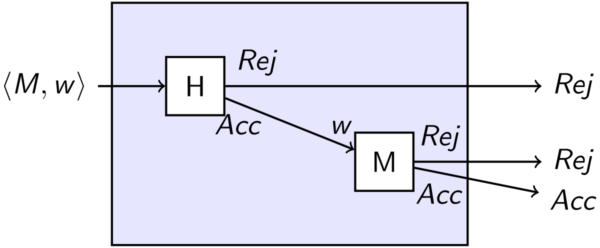
This proof strategy is called a reduction.

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?

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

Some more undecidable problems

The regularity problem for TMs

Does a given Turing machine accept a regular language?

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

- $L_{TM}^{REG} = \{\langle M \rangle \mid M$ 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.

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*.

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 $L(M_1) = L(M_2)\}$.

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: Reduce to emptiness.

Which is easier: Emptiness or non-emptiness?

- $L_e = \{\langle M \rangle \mid L(M) = \emptyset\}$
- $L_{ne} = \{\langle M \rangle \mid L(M) \neq \emptyset\}$

Is everything about Turing machines undecidable?

1. $\{\langle M \rangle \mid L(M) = \emptyset\}$

Is everything about Turing machines undecidable?

1. $\{\langle M\rangle \mid L(M) = \emptyset\}$
2. $\{\langle M\rangle \mid L(M) \text{ is regular }\}$

Is everything about Turing machines undecidable?

1. $\{\langle M \rangle \mid L(M) = \emptyset\}$
2. $\{\langle M \rangle \mid L(M) \text{ is regular } \}$
3. $\{\langle M \rangle \mid L(M) \text{ is context-free } \}$

Is everything about Turing machines undecidable?

1. $\{\langle M \rangle \mid L(M) = \emptyset\}$
2. $\{\langle M \rangle \mid L(M)$ is regular $\}$
3. $\{\langle M \rangle \mid L(M)$ is context-free $\}$
4. $\{\langle M \rangle \mid L(M)$ is finite $\}$

Is everything about Turing machines undecidable?

1. $\{\langle M \rangle \mid L(M) = \emptyset\}$
2. $\{\langle M \rangle \mid L(M)$ is regular $\}$
3. $\{\langle M \rangle \mid L(M)$ is context-free $\}$
4. $\{\langle M \rangle \mid L(M)$ is finite $\}$
5. $\{\langle M \rangle \mid M$ has 5 states $\}$

Is everything about Turing machines undecidable?

1. $\{\langle M\rangle \mid L(M) = \emptyset\}$
2. $\{\langle M\rangle \mid L(M)$ is regular $\}$
3. $\{\langle M\rangle \mid L(M)$ is context-free $\}$
4. $\{\langle M\rangle \mid L(M)$ is finite $\}$
5. $\{\langle M\rangle \mid M$ has 5 states $\}$
6. $\{\langle M\rangle \mid L(M)$ is a language $\}$

Is everything about Turing machines undecidable?

1. $\{\langle M\rangle \mid L(M) = \emptyset\}$
2. $\{\langle M\rangle \mid L(M)$ is regular $\}$
3. $\{\langle M\rangle \mid L(M)$ is context-free $\}$
4. $\{\langle M\rangle \mid L(M)$ is finite $\}$
5. $\{\langle M\rangle \mid M$ has 5 states $\}$
6. $\{\langle M\rangle \mid L(M)$ is a language $\}$
7. $\{\langle M\rangle \mid M$ makes at least 5 moves on some input$\}$

Is everything about Turing machines undecidable?

1. $\{\langle M \rangle \mid L(M) = \emptyset\}$
2. $\{\langle M \rangle \mid L(M)$ is regular $\}$
3. $\{\langle M \rangle \mid L(M)$ is context-free $\}$
4. $\{\langle M \rangle \mid L(M)$ is finite $\}$
5. $\{\langle M \rangle \mid M$ has 5 states $\}$
6. $\{\langle M \rangle \mid L(M)$ is a language $\}$
7. $\{\langle M \rangle \mid M$ makes at least 5 moves on some input$\}$

Rice's Theorem

Is everything about Turing machines undecidable?

1. $\{\langle M \rangle \mid L(M) = \emptyset\}$
2. $\{\langle M \rangle \mid L(M)$ is regular $\}$
3. $\{\langle M \rangle \mid L(M)$ is context-free $\}$
4. $\{\langle M \rangle \mid L(M)$ is finite $\}$
5. $\{\langle M \rangle \mid M$ has 5 states $\}$
6. $\{\langle M \rangle \mid L(M)$ is a language $\}$
7. $\{\langle M \rangle \mid M$ makes at least 5 moves on some input$\}$

Rice's Theorem

Any "non-trivial" property of R.E languages is undecidable!

Rice's theorem

Rice's theorem (1953)

Any non-trivial property of R.E languages is undecidable!

- Property $P \equiv$ set of languages (i.e., their TM encodings) satisfying P
- Property of r.e languages: membership of M in P depends only on the language of M. If $L(M) = L(M')$, then $\langle M \rangle \in P$ iff $\langle M' \rangle \in P$.
- Non-trivial: It holds for some but not all TMs.