Computability and Complexity

Lecture 3

Decidable and undecidable problems from formal languages
Closure properties of decidable languages
Closure properties of recognizable languages

given by Jiri Srba

Problems from Formal Language Theory

Decision Problems

- Acceptance: does a given string belong to a given language?
- Emptiness: is a given language empty?
- Equality: are given two languages equal?

These problems make sense only if we specify how the given languages are described (they must have a finite description e.g. via finite automata, context-free grammars or Turing machines).

Acceptance Problem for DFA

Problem: "Given a DFA B and a string w, does B accept w?

Language Formulation (Acceptance Problem for DFA)

$$A_{DFA} \stackrel{\text{def}}{=} \{\langle B, w\rangle \mid \ B \text{ is a DFA, } w \text{ a string and } B \text{ accepts } w\}$$

Acceptance Problem for DFA

Problem: "Given a DFA B and a string w, does B accept w?

Language Formulation (Acceptance Problem for DFA)

$$A_{DFA} \stackrel{\text{def}}{=} \{\langle B, w\rangle \mid \text{ } B \text{ is a DFA, } w \text{ a string and } B \text{ accepts } w\}$$

Theorem

The language A_{DFA} is decidable.

Proof: Construct a decider M for A_{DFA}:

Acceptance Problem for DFA

Problem: "Given a DFA B and a string w, does B accept w?

Language Formulation (Acceptance Problem for DFA)

$$A_{DFA} \stackrel{\text{def}}{=} \{\langle B, w\rangle \mid \text{ } B \text{ is a DFA, } w \text{ a string and } B \text{ accepts } w\}$$

Theorem

The language A_{DFA} is decidable.

Proof: Construct a decider M for A_{DFA}:

$M =$ " On input x:

1. Check if x is of the form $\langle B, w\rangle$ where B is an DFA and w is a string, if not then M rejects.
2. Simulate B on w (states of B are stored on a tape).
3. If the simulation accepted then M accepts. If it rejected then M rejects."

Acceptance Problem for NFA

Problem: "Given a NFA B and a string w, does B accept w?

Language Formulation (Acceptance Problem for NFA)

$$A_{NFA} \stackrel{\mathrm{def}}{=} \{\langle B, w\rangle \mid \text{ } B \text{ is an NFA, } w \text{ a string and } B \text{ accepts } w\}$$

Acceptance Problem for NFA

Problem: "Given a NFA B and a string w, does B accept w?

Language Formulation (Acceptance Problem for NFA)

$$A_{NFA} \stackrel{\text{def}}{=} \{\langle B, w\rangle \mid B \text{ is an NFA, } w \text{ a string and } B \text{ accepts } w\}$$

Theorem

The language A_{NFA} is decidable.

Proof: Construct a decider M for A_{NFA}:

Acceptance Problem for NFA

Problem: "Given a NFA B and a string w, does B accept w?

Language Formulation (Acceptance Problem for NFA)

$$A_{NFA} \stackrel{\text{def}}{=} \{\langle B, w\rangle \mid \ B \text{ is an NFA, } w \text{ a string and } B \text{ accepts } w\}$$

Theorem

The language A_{NFA} is decidable.

Proof: Construct a decider M for A_{NFA}:

$M =$ " On input x:

1. Check if x is of the form $\langle B, w\rangle$ where B is an NFA and w is a string, if not then M rejects.
2. Convert B to an equivalent DFA B'.
3. Run the algorithm for A_{DFA} on B' and w."

Emptiness Problem for NFA (and DFA)

Problem: "Given a NFA A is the language $L(A)$ empty?

Language Formulation (Emptiness Problem for NFA)

$$E_{NFA} \stackrel{\text{def}}{=} \{\langle A\rangle \mid \ A \text{ is an NFA and } L(A) = \emptyset \ \}$$

Emptiness Problem for NFA (and DFA)

Problem: "Given a NFA A is the language $L(A)$ empty?

Language Formulation (Emptiness Problem for NFA)

$$E_{NFA} \stackrel{\text{def}}{=} \{\langle A\rangle \mid \ A \text{ is an NFA and } L(A) = \emptyset \ \}$$

Theorem

The language E_{NFA} (as well as E_{DFA}) is decidable.

Proof: Construct a decider M for E_{NFA}:

Emptiness Problem for NFA (and DFA)

Problem: "Given a NFA A is the language $L(A)$ empty?

Language Formulation (Emptiness Problem for NFA)

$$E_{NFA} \stackrel{\text{def}}{=} \{\langle A\rangle \mid \ A \text{ is an NFA and } L(A) = \emptyset \ \}$$

Theorem

The language E_{NFA} (as well as E_{DFA}) is decidable.

Proof: Construct a decider M for E_{NFA}:

$M =$ " On input x, check if x is of the form $\langle A\rangle$, if not M rejects.

2. Mark all accept states of A.
3. Repeat until no new states are marked:
   Mark any state which has a transition to
   an already marked state.
4. If the start state is marked, then M rejects, else accepts."

Equality Problem for NFA (and DFA)

Problem: "Given two NFA A and B is $L(A)$ equal to $L(B)$"?

Language Formulation (Equality Problem for NFA)

$$EQ_{NFA} \stackrel{\text{def}}{=} \{\langle A, B\rangle \mid \ A \text{ and } B \text{ are NFA and } L(A) = L(B) \ \}$$

Equality Problem for NFA (and DFA)

Problem: "Given two NFA A and B is $L(A)$ equal to $L(B)$"?

Language Formulation (Equality Problem for NFA)

$$EQ_{NFA} \stackrel{\text{def}}{=} \{\langle A, B\rangle \mid \ A \text{ and } B \text{ are NFA and } L(A) = L(B) \ \}$$

Theorem

The language EQ_{NFA} (as well as EQ_{DFA}) is decidable.

Proof: Construct a decider M for EQ_{NFA}:

Equality Problem for NFA (and DFA)

Problem: "Given two NFA A and B is $L(A)$ equal to $L(B)$"?

Language Formulation (Equality Problem for NFA)

$$EQ_{NFA} \stackrel{\text{def}}{=} \{\langle A, B\rangle \mid \ A \text{ and } B \text{ are NFA and } L(A) = L(B)\ \}$$

Theorem

The language EQ_{NFA} (as well as EQ_{DFA}) is decidable.

Proof: Construct a decider M for EQ_{NFA}:

$M =$ " 1. On input x, check if $x = \langle A, B\rangle$, if not $\underline{M \text{ rejects}}$.
2. Construct NFA C for the language $\big(L(A) \cap \overline{L(B)}\big) \cup \big(\overline{L(A)} \cap L(B)\big)$.
4. Check if $L(C) = \emptyset$, if yes then $\underline{M \text{ accepts}}$, else $\underline{\text{rejects}}$."

Equality Problem for NFA (and DFA)

Problem: "Given two NFA A and B is $L(A)$ equal to $L(B)$"?

Language Formulation (Equality Problem for NFA)

$$EQ_{NFA} \stackrel{\text{def}}{=} \{\langle A, B\rangle \mid \text{ } A \text{ and } B \text{ are NFA and } L(A) = L(B) \text{ }\}$$

Theorem

The language EQ_{NFA} (as well as EQ_{DFA}) is decidable.

Proof: Construct a decider M for EQ_{NFA}:

$M =$ "

1. On input x, check if $x = \langle A, B\rangle$, if not $\underline{M \text{ rejects}}$.
2. Construct NFA C for the language $\big(L(A) \cap \overline{L(B)}\big) \cup \big(\overline{L(A)} \cap L(B)\big)$.
4. Check if $L(C) = \emptyset$, if yes then $\underline{M \text{ accepts}}$, else $\underline{\text{rejects}}$."

We reduced the equivalence problem to the emptiness problem!

Facts about Context-Free Grammars (CFG)

Chomsky Normal Form

A CFG is in Chomsky normal form if

- the rules are of the form $A \to BC$ or $A \to a$,
- the rule $S \to \epsilon$ is allowed, and
- no rule has S on the right-hand side.

Fact: every grammar can be converted to Chomsky normal form.

Lemma

If G is a context-free grammar in Chomsky normal form then any $w \in L(G)$ such that $w \neq \epsilon$ can be derived from S in exactly $2|w| - 1$ steps.

Acceptance Problem for CFG

Problem: "Given a CFG G and a string w, does G generate w?

Language Formulation (Acceptance Problem for CFG)

$$A_{CFG} \stackrel{\text{def}}{=} \{\langle G, w\rangle \mid \; G \text{ is a CFG, } w \text{ a string and } w \in L(G)\}$$

Theorem

The language A_{CFG} is decidable.

Proof: Construct a decider M for A_{CFG}:

Acceptance Problem for CFG

Problem: "Given a CFG G and a string w, does G generate w?

Language Formulation (Acceptance Problem for CFG)

$$A_{CFG} \stackrel{\text{def}}{=} \{\langle G, w\rangle \mid \ G \text{ is a CFG, } w \text{ a string and } w \in L(G)\}$$

Theorem

The language A_{CFG} is decidable.

Proof: Construct a decider M for A_{CFG}:

$M =$ "

1. On input x check if $x = \langle G, w\rangle$ where G is an CFG and w is a string, if not then <u>M rejects</u>.
2. Convert G into Chomsky normal form.
3. List all derivations in G of length exactly $2|w| - 1$, if $w = \epsilon$ then check if there is the rule $S \to \epsilon$.
4. If w is ever generated then <u>M accepts</u>, else <u>M rejects.</u>"

Emptiness Problem for CFG

Problem: "Given a CFG G, is $L(G)$ empty?

Language Formulation (Emptiness Problem for CFG)

$$E_{CFG} \stackrel{\text{def}}{=} \{\langle G \rangle \mid \; G \text{ is a CFG and } L(G) = \emptyset\}$$

Theorem

The language E_{CFG} is decidable.

Proof: Construct a decider M for E_{CFG}:

Emptiness Problem for CFG

Problem: "Given a CFG G, is $L(G)$ empty?

Language Formulation (Emptiness Problem for CFG)

$$E_{CFG} \stackrel{\text{def}}{=} \{\langle G\rangle \mid \ G \text{ is a CFG and } L(G) = \emptyset\}$$

Theorem

The language E_{CFG} is decidable.

Proof: Construct a decider M for E_{CFG}:

$M =$ "
1. On input x check if $x = \langle G\rangle$ where G is an CFG
2. Convert G into Chomsky normal form.
3. Mark all nonterminals A which have some rule $A \to a$.
4. Repeat until no new nonterminals are marked:
   Mark the nonterminal A if there is a rule
   $A \to BC$ such that B and C are already marked.
5. If S is marked ($L(G) \neq \emptyset$) then <u>M rejects</u>, else <u>accepts</u>.

Summary of Decidable (Undecidable) Problems

	Acceptance	Emptiness	Equality
DFA	yes	yes	yes
NFA	yes	yes	yes
CFG	yes	yes	no
TM	no	no	no

Closure Properties of Decidable Languages

Theorem (Closure Properties of Decidable Languages)

The class of decidable languages is closed under intersection, union, complement, concatenation and Kleene star.

Closure Properties of Decidable Languages

Theorem (Closure Properties of Decidable Languages)

The class of decidable languages is closed under intersection, union, complement, concatenation and Kleene star.

Q: What does closed under mean?

A: If L_1 and L_2 are decidable languages, then $L_1 \cap L_2$, $L_1 \cup L_2$, $\overline{L_1}$, $L_1.L_2$ and L_1^* are decidable too.

Proof: Closure of Decidable Languages under Intersection

Let L_1 and L_2 be decidable. We show that $L_1 \cap L_2$ is decidable too.

Let M_1 be a decider for L_1 and M_2 be a decider for L_2.

Consider a 2-tape TM M:

$M=$ "On input x:

1. copy x on the second tape
2. on the first tape run M_1 on x
3. if M_1 accepted then goto 4. else M <u>rejects</u>
4. on the second tape run M_2 on x
5. if M_2 accepted then M <u>accepts</u> else M <u>rejects</u>."

Proof: Closure of Decidable Languages under Intersection

Let L_1 and L_2 be decidable. We show that $L_1 \cap L_2$ is decidable too.

Let M_1 be a decider for L_1 and M_2 be a decider for L_2.

Consider a 2-tape TM M:

M= "On input x:
1. copy x on the second tape
2. on the first tape run M_1 on x
3. if M_1 accepted then goto 4. else M <u>rejects</u>
4. on the second tape run M_2 on x
5. if M_2 accepted then M <u>accepts</u> else M <u>rejects</u>."

- The machine M is a decider and it accepts a string x iff both M_1 and M_2 accept x.
- Two-tape TM is as expressive as the single tape TM.

□

Proof: Closure of Decidable Languages under Complement

Let L_1 be a decidable language. We show that $\overline{L_1}$ is decidable too.

Let M_1 be a decider for L_1.

Consider a TM M:

M= "On input x:
1. run M_1 on x
2. if M_1 accepted then M <u>rejects</u> else M <u>accepts</u>."

- The machine M is a decider, and it accepts a string x iff M_1 rejects x.
- Hence M decides $\overline{L_1}$.

$\square$

Closure Properties of Recognizable Languages

Theorem (Closure Properties of Recognizable Languages)

The class of recognizable languages is closed under intersection, union, concatenation and Kleene star.

BUT

Recognizable languages are not closed under complement!

Proof: Closure of Recognizable Languages under $\cup$

Let L_1 and L_2 be recognizable languages with the corresponding recognizers M_1 and M_2. We construct a recognizer M for $L_1 \cup L_2$.

Proof: Closure of Recognizable Languages under $\cup$

Let L_1 and L_2 be recognizable languages with the corresponding recognizers M_1 and M_2. We construct a recognizer M for $L_1 \cup L_2$.

Strategy I: run M_1 and M_2 in parallel on a 2-tape TM M

$M =$ "On input x:

1. Copy x on the second tape.
2. Do one step of M_1 on tape 1 and one step of M_2 on tape 2.
3. If either M_1 or M_2 accepted, then M accepts, else goto 2."

Proof: Closure of Recognizable Languages under $\cup$

Let L_1 and L_2 be recognizable languages with the corresponding recognizers M_1 and M_2. We construct a recognizer M for $L_1 \cup L_2$.

Strategy I: run M_1 and M_2 in parallel on a 2-tape TM M

$M =$ "On input x:

1. Copy x on the second tape.
2. Do one step of M_1 on tape 1 and one step of M_2 on tape 2.
3. If either M_1 or M_2 accepted, then M accepts, else goto 2."

Strategy II: nondeterministically choose to run M_1 or M_2

$M =$ "On input x:

1. Nondeterministically choose $i \in \{1, 2\}$.
2. Run machine M_i on the input x.
3. If M_i accepted, then M accepts.
   If M_i rejected, then M rejects."

Exam Questions

- Decidable problems on DFA and NFA (acceptance, emptiness, equality).
- Decidable problems on CFG (acceptance, emptiness).
- Closure properties of decidable languages.
- Closure properties of recognizable languages.