INT201 Decision, Computation and Language

Lecture 11 – Decidable Languages

Dr Yushi Li

Xi’an Jiaotong-Liverpool University
西交利物浦大学

Decidability

Is everything computable?

The answer is “No”, and most problems are not solvable by TMs and, therefore, not solvable by computers.

Why study decidability?

Some certain problems are unsolvable by TMs (computers). We should be able to recognize them.

Which languages are Turing decidable, Turing-recognizable or neither?

Decidability

Definition

Let be an alphabet and let $L \subseteq \Sigma^*$ be a language. We say that L is **decidable**, if there exists a Turing machine M, such that for every string $w \in \Sigma^*$, the following holds:

- If $w \in L$, then the computation of the Turing machine M, on the input string w, terminates in the accept state.
- If $w \notin L$, then the computation of the Turing machine M, on the input string w, terminates in the reject state.

Decidability

Given a language L whose elements are pairs of the form (B, w), where

- B is some computation model (e, g. DFA, NFA…).
- w is a string over the alphabet Σ.

The pair $(B, w) \in L$ if and only if $w \in L$.

Since the input to computation model B is a string over Σ, we must encode the pair (B, w) as a string.

Acceptance problem for computation model

Decision problem: Dose a given model accept/generate a given string w?

Instance $\langle B, w \rangle$ is the encoding of the pair (B, w).

Universe Ω comprises every possible instance:

$$\Omega = \{\langle B, w \rangle \mid B \text{ is a model and } w \text{ is a string}\}$$

Language comprises all "yes" instances

$$L = \{\langle B, w \rangle \mid B \text{ is a model that accept } w\} \subseteq \Omega$$

Acceptance problem for Language L_{DFA}

Decision problem: Dose a given DFA B accept a given string w?

Instance $\langle B, w\rangle$ is the encoding of the pair (B, w).

Universe Ω comprises every possible instance:

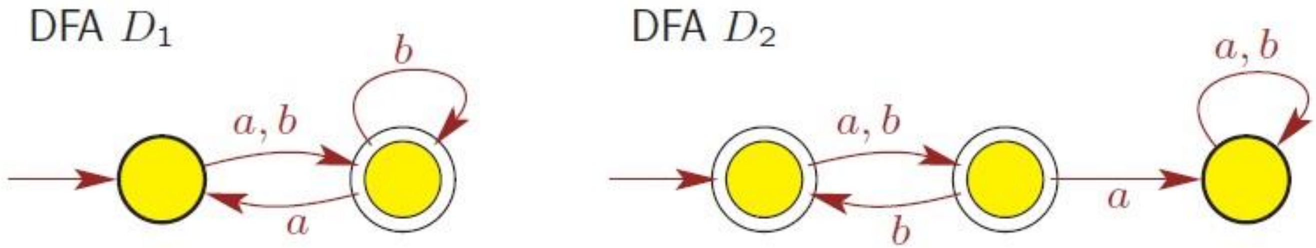
$$\Omega = \{\langle B, w\rangle \mid B \text{ is a DFA and } w \text{ is a string}\}$$

Language comprises all "yes" instances

$$L = \{\langle B, w\rangle \mid B \text{ is a DFA that accept } w\} \subseteq \Omega$$

Acceptance problem for Language L_{DFA}

Example



The Language L_{DFA} is decidable

$$L_{DFA} = \{\langle B, w\rangle \mid B \text{ is a DFA that accept } w\} \subseteq \Omega$$

$$\Omega = \{\langle B, w\rangle \mid B \text{ is a DFA and } w \text{ is a string}\}$$

To prove L_{DFA} is decidable, we need to construct TM M that decides L_{DFA}.

For M that decides L_{DFA} :

- take $\langle B, w\rangle \in \Omega$ as input
- halt and **accept** if $\langle B, w\rangle \in L_{DFA}$
- halt and **reject** if $\langle B, w\rangle \notin L_{DFA}$

The Language L_{DFA} is decidable

Proof

Basic idea:

On input $\langle B, w \rangle \in \Omega$, where

- $B = (\Sigma, Q, \delta, q_0, F)$ is a DFA
- $w = w_1 w_2 \cdots w_n \in \Sigma^*$ is input string to process on B.

1. Check if $\langle B, w \rangle$ is "proper" encoding. If not, reject
2. Simulate B on w based on:

- $q \in Q$, the current state of B
- $i \in \{1, 2, \ldots, |w|\}$, the pointer that illustrates the current position in w.
- q changes in accordance with w_i and the transition function $\delta(q, w_i)$.

3. If B ends in $q \in F$, then M accepts; otherwise, reject.

The Language L_{NFA} is decidable

Decision problem: Dose a given NFA B accept a given string w?

$$L_{NFA} = \{\langle B, w\rangle \mid B \text{ is a NFA that accept } w\} \subseteq \Omega$$

$$\Omega = \{\langle B, w\rangle \mid B \text{ is a NFA and } w \text{ is a string}\}$$

Proof

On input $\langle B, w\rangle \in \Omega$, where

- $B = (\Sigma, Q, \delta, q_0, F)$ is a NFA
- $w \in \Sigma^*$ is input string to process on B.

The Language L_{NFA} is decidable

Proof

The Language L_{REX} is decidable

Decision problem: Dose a regular expression R generate a given string w?

$$L_{REX} = \{\langle R, w\rangle \mid R \text{ is a regular expression that generates } w\} \subseteq \Omega$$

$$\Omega = \{\langle R, w\rangle \mid R \text{ is regular expression and } w \text{ is a string}\}$$

Example

Given regular expression R = aa*b

$$\langle R, aab\rangle \in L_{REX},\ \langle R, aba\rangle \notin L_{REX}$$

The Language L_{REX} is decidable

Proof

On input $\langle R, w \rangle \in \Omega$

CFGs are decidable

Decision problem: Dose a CFG G generate a string w ?

$$L_{CFG} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\} \subseteq \Omega$$

$$\Omega = \{\langle G, w\rangle \mid G \text{ is a CFG and } w \text{ is a string}\}$$

$\langle G, w\rangle \in L_{CFG}$ if G generates w, $w \in L(G)$

$\langle G, w\rangle \notin L_{CFG}$ if G dosen't generate w, $w \notin L(G)$

CFGs are decidable

Recall

A context-free grammar $G = (V, \Sigma, R, S)$ is in **Chomsky normal form** if each rule is of the form

$$A \to BC \text{ or } A \to x \text{ or } S \to \varepsilon$$

- variable $A \in V$
- variables $B, C \in V - \{S\}$
- terminal $x \in \Sigma$.

Every CFG can be converted into Chomsky normal form

CFG G in Chomsky normal form is easier to analyze.

- Can show that for any string $w \in L(G)$ with $w \neq \varepsilon$ by derivation $S \overset{*}{\Rightarrow} w$ takes exactly $2|w| - 1$ steps.
- $\varepsilon \in L(G)$ if G includes rule $S \to \varepsilon$.

CFGs are decidable

Proof

CFLs are decidable

Every CFL L is a decidable language.

Proof

Tutorial

The Language L_{TM}

The language L_{TM} is undecidable.

$$L_{TM} = \{\langle M, w\rangle : M \text{ is a Turing machine that accepts the string } w\}$$

$$\Omega = \{\langle M, w\rangle | \ G \text{ is a TM and } w \text{ is a string}\}$$

- If M accepts w, then $\langle M, w\rangle \in L_{TM}$
- If M doesn't accept w (reject or loop), then $\langle M, w\rangle \notin L_{TM}$

Proof

Tutorial

Unsolvable and undecidable problems

Definition

Undecidable problem. The associated language of a problem cannot be recognized by a TM that halts for all inputs. (one problem that should give a "yes" or "no" answer, but yet no algorithm exists that can answer correctly on all inputs.)

Unsolvable problem. A computational problem that cannot be solved by a TM.

Undecidable problem is a subcategory of Unsolvable problem.

Example

Halting problem is to determine for an arbitrary TM M and an arbitrary input string w whether M with input w halts or not.

$$L_{TM} = \{\langle M, w\rangle : M \text{ is a Turing machine that terminates on the input string } w\}$$

Countable set

Let A and B be two sets and let $f : A \to B$ be a function. Recall that f is called a bijection, if

- f is one-to-one (or injective), i.e., for any two distinct elements a and a' in A, we have $f(a) \neq f(a')$,
- f is onto (or surjective), i.e., for each element $b \in B$, there exists an element $a \in A$, such that $f(a) = b$.

The set of natural numbers is denoted by N. That is, $N = \{1, 2, 3, \ldots\}$.

Definition

Let A and B be two sets. We say that A and B have the **same size**, if there exists a bijection $f: A \to B$.

Let A be a set. We say that A is **countable**, if A is finite, or A and N have the same size.

Countable set

Example

The set $\mathbb{Z}$ of integers: $\mathbb{Z} = \{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$ is countable

Proof

To prove that the set $\mathbb{Z}$ is countable, we have to give each element of $\mathbb{Z}$ a unique number in $\mathbb{N}$. We obtain this numbering, by listing the elements of $\mathbb{Z}$ in the following order:

$$0, 1, -1, 2, -2, 3, -3, 4, -4, \ldots$$

In this (infinite) list, every element of $\mathbb{Z}$ occurs exactly once. The number of an element of $\mathbb{Z}$ is given by its position in this list. Formally, define the function $f : \mathbb{N} \to \mathbb{Z}$ by

$f(n) = n/2$ if n is even, $-(n-1)/2$ if n is odd.

This function f is a bijection and, therefore, the sets $\mathbb{N}$ and $\mathbb{Z}$ have the same size. Hence, the set $\mathbb{Z}$ is countable.

Countable set

Example

The set $\mathbb{Q}$ of rational numbers: $\mathbb{Q} = \{m/n : m \in \mathbb{Z}, n \in \mathbb{Z}, n \neq 0\}$ is countable

Proof

$$
\begin{array}{cccccc}
1/1 & 1/2 & 1/3 & 1/4 & 1/5 & \cdots \\
2/1 & 2/2 & 2/3 & 2/4 & 2/5 & \cdots \\
3/1 & 3/2 & 3/3 & 3/4 & 3/5 & \cdots \\
4/1 & 4/2 & 4/3 & 4/4 & 4/5 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{array}
$$

If we try to first list all elements in first row, then list all elements in second row, and so on, then we will never get to the second row because the first row is infinitely long.

Countable set

Example

Proof

Instead, we can enumerate elements using clever method

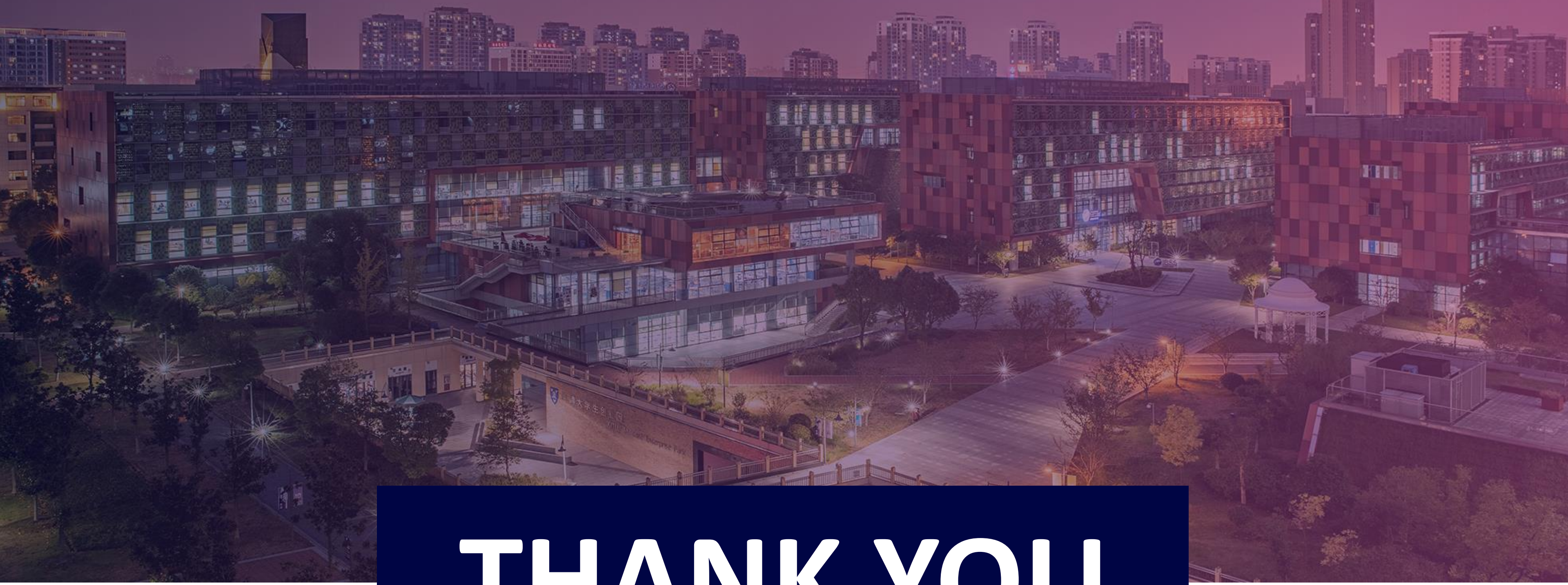
$$
\begin{array}{cccccc}
1/1 & 1/2 & 1/3 & 1/4 & 1/5 & \cdots \\
2/1 & 2/2 & 2/3 & 2/4 & 2/5 & \cdots \\
3/1 & 3/2 & 3/3 & 3/4 & 3/5 & \cdots \\
4/1 & 4/2 & 4/3 & 4/4 & 4/5 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{array}
$$

Uncountable set

Definition

A set is **uncountable** if it contains so many elements that there is no bijection between this set and the set of natural numbers (N).

Example. The set R of all real numbers is uncountable



THANK YOU

Xi'an Jiaotong-Liverpool University
西交利物浦大学

XJTLU | SCHOOL OF FILM AND TV ARTS