可计算理论

Theory of Computability

刘显敏
liuxianmin@hit.edu.cn

海量数据计算研究中心
HARBIN INSTITUTE OF TECHNOLOGY

计算理论—2023年春

为问题设计算法的难度不同

2

为如下问题设计算法
- 设计算法判断输入字符串是否属于某个正则语言;
- 设计算法判断两个DFA是否识别同一个正则语言;
- 设计算法判断输入是否属于某个上下文无关语言;
- 设计算法判断两个上下文无关文法是否生成相同语言;
- 设计算法完成字符串搜索任务;
- 设计算法求解波斯特对应问题;

3

如何定义可计算?

可判定(decidable)
可识别(recognizable)

可判定 vs 可识别

4

- 回忆一下图灵机语言的定义

定义 图灵机M的语言(M识别的语言)
M接受的字符串集合成为M的语言，或被M识别的语言，记作$L(M)$。

- $w \in L(M) \Leftrightarrow$ 输入w图灵机M会进入接受状态
- 那么如何知道$w \notin L(M)$呢? M可能会不停机

定义 图灵机M的语言(M判定的语言)
如果图灵机M是**始终停机**的，那么称$L(M)$为图灵机M判定的语言。

- 始终停机重要吗?
- 停机的才是算法

可判定 vs 可识别

5

- 可计算性是“谁的可计算性”？
- 两个问题:
  - 一个图灵机对应几个语言?
  - 一个语言对应几个图灵机?

定义 图灵可识别语言，Turing-Recognizable Language
如果存在图灵机M识别语言L，那么L是一个图灵可识别语言，即Turing-recognizable

定义 图灵可判定语言，Turing-Decidable Language
如果存在图灵机M判定语言L，那么L是一个图灵可判定语言，即Turing-decidable

- 两种可计算性的定义，对应两种“可解程度”

可判定 vs 可识别

6

定理 判定 vs 识别
如果图灵机M判定L，那么M一定识别L。

定理 可判定 vs 可识别
如果语言L是图灵可判定的，L一定也是图灵可识别的。

- 如何证明某个语言（问题）是图灵可判定的?
  - 设计算法
- 例如：每一个正则语言L都是可判定的。
- 存在语言L，L是图灵可识别的，但不是图灵可判定的
  - 如何证明?
- 存在语言L，L不是图灵可识别的

区分问题与实例

7

- 问题-problem；实例-instance
- 语言-language；字符串-string
- 可计算理论关注的是：具有无穷多输入实例的问题到底多难以计算（是否有算法）！

例 如何讨论如下猜想的计算难度

对比考虑如下程序（$3X+1$猜想）

```
while x ≠ 1 do
    if x is odd then x = 3x + 1;
    else x = x/2;
return;
```

```
if 3X + 1猜想为真 then
    return 1;
else
    return 0;
```

令A是可以令上述程序停机的自然数集合。

A是否是图灵可识别的?

A是否是图灵可判定的?

8

如何定义可计算?

递归的(recursive)

递归可枚举的(recursively enumerable)

递归可枚举 vs 递归

9

- 图灵机的另一种形式

定义 枚举图灵机(非形式化)

枚举图灵机(enumerator)是一个多带图灵机：

- 初始状态下，工作带是空的，即无输入
- 有一条特殊输出带，带头只能右移，在输出带上顺序打印有穷长度的字符串，字符串的数目可能是无穷的，字符串可以重复打印
- 按照如下方式工作：
  - 运行中会进入一个特殊的状态q_{print}，该状态将工作带上的内容复制到输出带上，利用#字符分隔
  - 复制后，机器返回常规状态继续运行

- 给定枚举图灵机E，对应语言为$L(E)=\{x|E\text{输出}\#x\#\}$

递归可枚举 vs 递归

10

定义 递归可枚举语言(recursively enumerable, r.e.)

令L为任一语言，如果存在枚举图灵机E满足$L=L(E)$，那么L是**递归可枚举语言**。

- 除非E一直运行，否则$L(E)$是有穷的
- 同一字符串可以被多次打印
- 字符串出现的顺序没有要求

定理 递归可枚举 vs 图灵可识别

语言L是递归可枚举的，当且仅当L是图灵可识别的。

- 不同领域，一个形式化语言的如下性质是等同的

recursively enumerable | (Turing) recognizable | Turing acceptable | partially decidable | semi-decidable

---

递归可枚举 vs 递归

11

- 如果要求字符串只被打印一次?

定理

如果语言L是递归可枚举的，那么存在一个枚举器E，满足$L(E)=L$且每个L中的字符串仅被输出一次

- 如果要求字符串以指定顺序（字母序）打印?

定义 递归语言(recursive)

令L为任一语言，如果存在枚举图灵机E满足$L=L(E)$，并且E以字母序打印$L(E)$，则称L是**递归语言**。

定理 递归 vs 图灵可判定

语言L是递归的，当且仅当L是图灵可判定的。

12

如何定义可计算?

递归 ⇔ 图灵可判定

递归可枚举 ⇔ 图灵可识别

一些重要的性质

13

已知问题(语言)A和B是可判定或可识别，然后呢？

定理 封闭性A

图灵可识别语言和图灵可判定语言在**集合并**以及**集合交**操作下是封闭的，即，如果A和B同为可判定(或可识别)语言，那么$A\cup B$和$A\cap B$是可判定(或可识别)的。

定理 封闭性B

图灵可识别语言和图灵可判定语言在**连接**和**闭包**操作下是封闭的，即，如果A和B同为可判定(或可识别)语言，那么如下两个语言也是可判定(或可识别)的。

- $C = A \cdot B = AB = \{xy \mid x \in A, y \in B\}$
- $S = A^* = \bigcup_{n=0}^{\infty} A^n$，其中$A^0 = \{\epsilon\}, A^n = AA^{n-1}$

一些重要的性质

14

已知问题(语言)A和B是可判定或可识别，然后呢？

定理 封闭性--补操作

图灵可判定语言在**集合补**操作下是封闭的，即，如果A是可判定语言，那么其补语言$\bar{A}$是可判定的。

- 图灵可识别语言在**补操作**下是否封闭？ **不封闭**

能否通过L和$\bar{L}$的识别图灵机构造算法 除非图灵可识别与图灵可判定等价

定理 图灵可判定 vs 图灵可识别

语言L是图灵可判定的，当且仅当L及其补语言$\bar{L}$均是图灵可识别的。

$\Rightarrow$ 显然

$\Leftarrow$ 同时模拟L和$\bar{L}$对应图灵机，直到其中一个停机

15

如何定义可计算？

图灵可判定 ☺

图灵可识别 ☹

问题(语言)分类

16

依据是否图灵可识别、可判定，一个具体的语言L可能出现如下**四种可能**

- **L和$\bar{L}$都是图灵可识别的，即都是图灵可判定的**
- **L是图灵可识别的，$\bar{L}$不是图灵可识别的**
- **$\bar{L}$是图灵可识别的，L不是图灵可识别的**
- **L和$\bar{L}$都不是图灵可识别的**

从是否存在求解算法的角度看

- **可计算⇔图灵可判定⇔递归语言**
- **不可计算⇔ 不可判定⇔ 非递归**

问题(语言)分类

17

- 真的存在图灵机不可识别的语言吗？
- 真的存在图灵机可识别但不可判定的语言吗？

一种计数方式的证明

- 一个图灵机只有一个语言
- 总共有多少不同的语言呢？
  - $|2^{\Sigma^*}|$，至少同实数一样多，不可数，基数$\aleph_1$
- 总共有多少不同的图灵机呢？
  - 图灵机编码
  - 每个图灵机至少与一个01字符串对应
  - 同自然数一样多，至多可数，基数$\aleph_0$
- 语言的数目远远多于图灵机的数量，因此，一定存在某个语言不对应任何一个图灵机

问题(语言)分类

18

图灵机编码

- 简化假设：处理{0,1}字符串，q_0为初始状态，q_1为接受状态

$$M = (Q, \{0,1\}, \{0,1,B\}, \delta, q_0, B, \{q_1\})$$

$Q = \{q_0, q_1, \ldots, q_n\}$，令$X_1 = 0, X_2 = 1, X_3 = B, D_1 = L,\ D_2 = R$

- 将$\delta(q_i, X_j) = (q_k, X_l, D_m)$编码为$0^{i+1}10^j10^{k+1}10^l10^m$
- 将图灵机M编码为(假设δ有r条转移规则)如下形式，这里$code_i$是δ中第i个转移规则的编码

$$\langle M\rangle = code_1 11 code_2 11 \ldots 11 code_r 111 \ldots$$

- 任意图灵机均可以编码为01字符串，若M的编码是第i个01字符串，称M是第i个图灵机，记作M_i
- 并非所有01字符串都是一个合法的图灵机编码

- 同自然数一样多，至多可数，基数$\aleph_0$
- 语言的数目远远多于图灵机的数量，因此，一定存在某个语言不对应任何一个图灵机

问题(语言)分类

19

图灵机编码

- 简化假设：处理{0,1}字符串，q_0为初始状态，q_1为接受状态

$$M = (Q, \{0,1\}, \{0,1,B\}, \delta, q_0, B, \{q_1\})$$

$Q = \{q_0, q_1, \ldots, q_n\}$，令$X_1 = 0, X_2 = 1, X_3 = B, D_1 = L$，$D_2 = R$

$$\delta(q_0, 1) = (q_2, 0, R) \quad \delta(q_2, 0) = (q_0, 1, R)$$
$$\delta(q_2, 1) = (q_1, 0, R) \quad \delta(q_2, B) = (q_2, 1, L)$$

$\langle M \rangle =$ 010010001010011000101010010011
000100100101001100010001000100101110 10101

- 任意图灵机均可以编码为01字符串，若M的编码是第i个01字符串，称M是第i个图灵机，记作M_i
- 并非所有01字符串都是一个合法的图灵机编码
  - 同自然数一样多，至多可数，基数$\aleph_0$
- 语言的数目远远多于图灵机的数量，因此，一定存在某个语言不对应任何一个图灵机

20

存在可识别但不可判定的语言吗
如何证明不可识别？
如何证明不可判定？

对角化语言L_d

21

- 令w_i表示Σ^*中按照字母序排列的第i个字符串
- 令M_i为w_i对应的图灵机
  - 如果w_i不是合法图灵机编码，$L(M_i) = \emptyset$

定义 对角化语言(L_d)

对角化语言L_d定义为$L_d = \{w_i | w_i \notin L(M_i)\}$。

定理 L_d

L_d不是图灵可识别的，进而，L_d不是图灵可判定的，不存在算法。

图灵机接受问题L_{ACC}

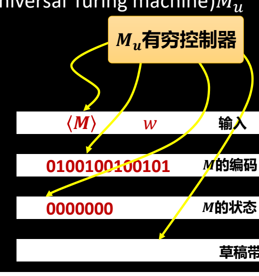
22

$L_{ACC} = \{\langle M, w \rangle | M$是一个图灵机，$M$接受$w\}$

定理 L_{ACC}是图灵可识别的。

证明思路 构造通用图灵机(universal Turing machine)M_u

- M_u输入$\langle M, w \rangle$
  1. 检查M的合法性
  2. 在w上模拟M运行，how?
  3. 若M接受w，则M_u接受
  4. 若M拒绝w，则M_u拒绝
- M_u接受$\langle M, w \rangle \Leftrightarrow M$接受$w$
- M_u识别L_{ACC}



图灵机接受问题L_{ACC}

23

$L_{ACC} = \{\langle M, w \rangle | M$是一个图灵机，$M$接受$w\}$

定理 L_{ACC}是图灵可识别的。

证明思路 构造通用图灵机(universal Turing machine)M_u

- M_u输入$\langle M, w \rangle$
  1. 检查M的合法性
  2.
  3.
  4.
- M_u接
- M_u识

M_u有穷控制器

输入

M的编码

M的状态

草稿带

❶M_u可适用于任意图灵机编码的处理
❷M_u模拟给定编码对应图灵机的行为
❸M_u实际上是一台支持编程的图灵机
❹M_u是现代的通用计算机的理论基础

图灵机接受问题L_{ACC}

24

$L_{ACC} = \{\langle M, w \rangle | M$是一个图灵机，$M$接受$w\}$

M_u是否可以判定L_{ACC}？

- 直观上M_u不能，是否存在别的图灵机可以判定L_{ACC}？

定理 L_{ACC}不可计算性

L_{ACC}不是图灵可判定的。

证明思路 利用对角线法构造悖论，假设判定L_{ACC}的机器H存在，构造不可能存在的机器D

$$H(\langle M, w \rangle) \Rightarrow H'(\langle M \rangle) \Rightarrow D(\langle M \rangle) \Rightarrow D(\langle D \rangle)$$

定理 推论

$\overline{L_{ACC}}$不是图灵可识别的。

图灵机停机问题L_{HALT}

25

$L_{HALT}=\{\langle M,w\rangle|M$是一个图灵机且$M$在$w$上停机$\}$

定理 L_{HALT}

L_{HALT}不是图灵可判定的。

证明思路

- 根据已有的结论，构造悖论
- 假设L_{HALT}可判定，利用M_{HALT}构造M_{ACC}判定L_{ACC}

定理 L_{HALT}

L_{HALT}是图灵可识别的。

定理 推论

$\overline{L_{HALT}}$不是图灵可识别的.

L_{ACC01}

26

$L_{ACC01}=\{\langle M\rangle|M$是图灵机且$M$接受$01\}$

定理 L_{ACC01}不是图灵可判定的。

证明思路

假设M_R是判定L_{ACC01}的图灵机，构造M_S判定L_{ACC}

M_S：输入$\langle M,w\rangle$，构造M'（输入x，模拟M在w上的运行，接受/拒绝若M接受/拒绝），模拟M_R在$\langle M'\rangle$上的运行，接受/拒绝若M_R接受/拒绝

- M_S接受/拒绝$\langle M,w\rangle \Leftrightarrow M_R$接受/拒绝$\langle M'\rangle$
- M_R接受/拒绝$\langle M'\rangle \Leftrightarrow M'$接受/拒绝$01$
- M'接受/拒绝$01 \Leftrightarrow M'$接受/拒绝$x \Leftrightarrow M$接受/拒绝w

定理 L_{ACC01}图灵可识别，$\overline{L_{ACC01}}$不是图灵可识别的。

L_e

27

$L_e=\{\langle M\rangle|M$是图灵机且不接受任何输入$\}$

定理 L_e

L_e不是图灵可判定的。

输入x，模拟M在w上的运行，接受/拒绝若M接受/拒绝

证明思路

假设令M_e是判定L_e的图灵机，构造M_S判定L_{ACC}

M_S：输入$\langle M,w\rangle$，构造M'，模拟M_e在$\langle M'\rangle$上的运行，接受/拒绝若M_e拒绝/接受；

- M_S接受/拒绝$\langle M,w\rangle \Leftrightarrow M_e$拒绝/接受$\langle M'\rangle$
- M_e拒绝/接受$\langle M'\rangle \Leftrightarrow M'$接受/拒绝所有字符串
- M'接受/拒绝所有字符串 $\Leftrightarrow M$接受/拒绝w

定理 L_e不是图灵可识别的，$\overline{L_e}$是图灵可识别的。

L_{REG}

28

$L_{REG}=\{\langle M\rangle|M$是接受正则语言的图灵机$\}$

定理 L_{REG}不是图灵可判定的。

证明思路 假设M_{REG}判定L_{REG}，构造M_S判定L_{ACC}

M_S:输入$\langle M,w\rangle$，

构造M'，输入x
如果x是0^n1^n，则接受；
否则模拟M在w上运行，接受/拒绝若M接受/拒绝

模拟M_{REG}在$\langle M'\rangle$上运行，接受/拒绝若M_{REG}接受/拒绝

- M_S接受/拒绝$\langle M,w\rangle \Leftrightarrow M_{REG}$接受/拒绝$\langle M'\rangle$
- M_{REG}接受/拒绝$\langle M'\rangle \Leftrightarrow M'$接受所有字符串/$L(M')=0^n1^n$
- M'接受所有字符串/$L(M')=0^n1^n \Leftrightarrow M$接受/拒绝$w$

L_{EQ}

29

$L_{EQ}=\{\langle M_1,M_2\rangle|L(M_1)=L(M_2)\}$

定理 L_{EQ}

L_{EQ}不是图灵可判定的。

证明思路 可以利用M_{EQ}构造判定L_e的图灵机

30

上述证明的思路几乎都是一样的

映射归约

31

归约(reduction) 支持将问题按 **计算难度** 分层(排序)

- 问题A可以归约到问题B，记作$\boldsymbol{A \le B}$
  - $A \le B$直观表示A不会比B更困难

映射(mapping)归约 | 多对一(many-to-one)归约 ...

定义 可计算函数(Computable Functions)

函数$f: \Sigma_1^* \to \Sigma_2^*$被称作是**可计算的**，若存在图灵机$M$使得$\forall w \in \Sigma_1^*, M$在$w$上停机且输出带上是$f(w)$

定义 映射归约(mapping reduction), $\le_m$

令$A \subseteq \Sigma_1^*$和$B \subseteq \Sigma_2^*$是两个语言，A被称作可映射归约(mapping-reducible)到B，记作$\boldsymbol{A \le_m B}$，若存在可计算函数$f: \Sigma_1^* \to \Sigma_2^*$使得$\forall w \in \Sigma_1^*$有$w \in A \Leftrightarrow f(w) \in B$。这里的函数$f$被称作从问题$A$到$B$的映射归约。

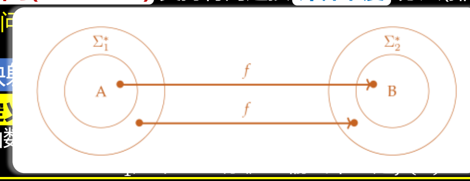
映射归约

32

归约(reduction) 支持将问题按 **计算难度** 分层(排序)

- 问

映 ...



定义

函数 ... 使得

定义 映射归约(mapping reduction), $\le_m$

令$A \subseteq \Sigma_1^*$和$B \subseteq \Sigma_2^*$是两个语言，A被称作可映射归约(mapping-reducible)到B，记作$\boldsymbol{A \le_m B}$，若存在可计算函数$f: \Sigma_1^* \to \Sigma_2^*$使得$\forall w \in \Sigma_1^*$有$w \in A \Leftrightarrow f(w) \in B$。这里的函数$f$被称作从问题$A$到$B$的映射归约。

$\le_m$的例子

33

例 $L_{ACC01} \le_m L_{ACC}$

- $f: \langle M \rangle \to \langle M, 01 \rangle$
- f定义了映射归约
  - 如果$\langle M \rangle \in L_{ACC01}$，那么$\langle M, 01 \rangle \in L_{ACC}$；
  - 如果$\langle M \rangle \notin L_{ACC01}$，那么$\langle M, 01 \rangle \notin L_{ACC}$；
  - f是可计算的函数；

例 $L_d \le_m \overline{L_{ACC}}$

- $f: \langle M \rangle \to \langle M, \langle M \rangle \rangle$
- f定义了映射归约
  - 如果$\langle M \rangle \in L_d$，那么$\langle M, \langle M \rangle \rangle \notin L_{ACC}$；
  - 如果$\langle M \rangle \notin L_d$，那么$\langle M, \langle M \rangle \rangle \in L_{ACC}$；
  - f是可计算函数；

$\le_m$的性质

34

定理 $\le_m$, I

如果$A \le_m B$且B是可判定的，那么A也是可判定的。

定理 $\le_m$, II

如果$A \le_m B$且A是不可判定的，那么B也不可判定。

定理 $\le_m$, III

如果$A \le_m B$且B是可识别的，那么A也是可识别的。

定理 $\le_m$,IV

如果$A \le_m B$且A是不可识别的，那么B也不可识别。

定理 $\le_m$, V

①$A \le_m B \Leftrightarrow A^c \le_m B^c$

②$A \le_m B, B \le_m C \Rightarrow A \le_m C$

$\le_m$的例子

35

例 $L_{ACC} \le_m L_{ACC01}$

- $f: \langle M, w \rangle \to \langle M'_{M,w} \rangle$
  - $M'_{M,w}$: 若M接受w, M'接受01；否则, 拒绝所有输入
- f定义了映射归约
  - $\langle M, w \rangle \in L_{ACC}$，那么$\langle M'_{M,w} \rangle \in L_{ACC01}$
  - $\langle M, w \rangle \notin L_{ACC}$，那么$\langle M'_{M,w} \rangle \notin L_{ACC01}$
  - f是可计算的函数

$\le_m$的例子

36

例 $L_{ACC} \le_m \overline{L_e}$

- $f: \langle M, w \rangle \to \langle M'_{M,w} \rangle$
- $M'_{M,\omega}$：如果M接受w，M'接受所有输入；否则，拒绝所有输入
- f定义了映射归约
  - $\langle M, w \rangle \in L_{ACC}$，那么$\langle M'_{M,w} \rangle \in \overline{L_e}$
  - $\langle M, w \rangle \notin L_{ACC}$，那么$\langle M'_{M,w} \rangle \in L_e$
  - f是可计算函数

$\le_m$的例子

37

例 $L_{ACC} \le_m L_{REG}$

- $f: \langle M,w\rangle \to \langle M'_{M,w}\rangle$
- $M'_{M,w}$：令M'默认接受0^n1^n，如果M接受w令M'接受；否则，令M'拒绝
- f定义了映射归约
  - $\langle M,w\rangle \in L_{ACC}$，那么$\langle M'_{M,w}\rangle \in L_{REG}$
  - $\langle M,w\rangle \notin L_{ACC}$，那么$\langle M'_{M,w}\rangle \notin L_{REG}$
  - f是可计算函数；

应用$\le_m$

38

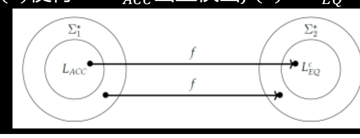
$$L_{EQ} = \{\langle M_1, M_2\rangle \mid L(M_1) = L(M_2)\}$$

定理 L_{EQ}

L_{EQ}不是图灵可识别的.

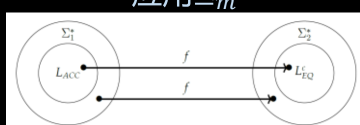
证明概要

- 证明$\overline{L_{ACC}} \le_m L_{EQ}$，已知$\overline{L_{ACC}}$不是图灵可识别的
- 等价地，需证明$L_{ACC} \le_m \overline{L_{EQ}}$
- 构造$f(x)$使得$x \in L_{ACC}$当且仅当$f(x) \in \overline{L_{EQ}}$



应用$\le_m$

39



- 如果x不是$\langle M,w\rangle$的合法输入，令$f(x) = \langle M_0, M_0\rangle$；此时，$x \notin L_{ACC}$且$f(x) \notin \overline{L_{EQ}}$
- 如果x形如$\langle M,w\rangle$，令$f(x) = \langle M_1, M_2\rangle$
  - M_1拒绝所有输入
  - M_2忽略输入，模拟M在w上运行，接受当且仅当M接受w
- 验证$x \in L_{ACC} \Leftrightarrow f(x) \in \overline{L_{EQ}}$
- $L_{ACC} \le_m \overline{L_{EQ}}$

应用$\le_m$

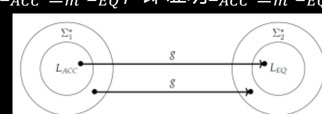
40

定理 $\overline{L_{EQ}}$

$\overline{L_{EQ}}$不是图灵可识别的。

证明概要

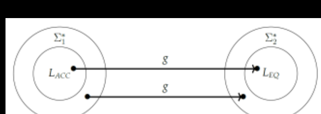
- 试证明$\overline{L_{ACC}} \le_m \overline{L_{EQ}}$，即证明$L_{ACC} \le_m L_{EQ}$



- 构造$g(x)$使得$x \in L_{ACC}$当且仅当$g(x) \in L_{EQ}$
- 如果x不是$\langle M,w\rangle$的合法输入，令$g(x) = \langle M_0, M_0'\rangle$并满足$L_{M_0} \neq L_{M_0'}$；此时，$x \notin L_{ACC}$且$g(x) \notin L_{EQ}$

应用$\le_m$

41



- 假设x形如$\langle M,w\rangle$，令$g(x) = \langle M_1, M_2\rangle$
  - M_1接受所有输入
  - M_2忽略输入模拟M在w上运行，M_2接受当且仅当M接受w
- 验证$x \in L_{ACC} \Leftrightarrow f(x) \in L_{EQ}$
- $L_{ACC} \le_m L_{EQ}$

图灵归约

42

- 图灵归约(Turing reduction)
- **神谕图灵机Oracle Turing Machine**
  - 一个图灵机变种，可以查询某个语言判定问题的结果是1或者0，M^B表示可以查询语言B的Oracle图灵机

定义 图灵归约, $\le_T$

令$A \subseteq \Sigma_1^*$和$B \subseteq \Sigma_2^*$是两个语言，A被称作可**图灵归约**(Turing-reducible)到B，记作$A \le_T B$，若存在Oracle图灵机M^B判定A。

定理 如果$A \le_m B$，那么$A \le_T B$。

按照如下方式构造M^B：

- 令M^B的输入为A的输入x；
- 计算$f(x)$，并且查询$f(x) \in B$?，返回查询的结果。

图灵归约

43

- 图灵归约(Turing reduction)

图灵归约比映射归约更为一般(general)

①$L_{ACC} \le_T \overline{L_{ACC}}$且$\overline{L_{ACC}} \le_T L_{ACC}$

②$L_{ACC} \le_m \overline{L_{ACC}}$?

③$\overline{L_{ACC}} \le_m L_{ACC}$?

定理 如果$A \le_m B$，那么$A \le_T B$。

按照如下方式构造M^B：

- 令M^B的输入为A的输入x；
- 计算$f(x)$，并且查询$f(x) \in B$?，返回查询的结果。

44

上述不可判定问题看起来**很相似**

Rice's Theorem

45

- 前述不可判定结论很多都是关于TM的性质
  - $L_{ACC01} = \{\langle M \rangle | M \text{ is a TM and } M \text{ accepts } 01\}$
  - $L_e = \{\langle M \rangle | M \text{ is a TM and } M \text{ accepts } nothing\}$
  - $L_{REG} = \{\langle M \rangle | M \text{ is a TM and } L(M) \text{ is regular}\}$
- 本质上，是**图灵机识别语言的性质**
- 对比一下图灵机的性质
  - $\{\langle M \rangle | M \text{ never tries to move left off the left end of the tape}\}$
  - $\{\langle M \rangle | M \text{ has more than 20 states }\}$
- 莱斯定理(Rice's Theorem)表明**任何图灵机识别语言的非平凡性质都是不可判定的**
- 一定是nontrivial properties

Rice's Theorem

46

要点：**非平凡性质(nontrivial properties)**

- P是图灵机识别语言的非平凡性质，当且仅当
  - 存在图灵机M_1使得$L(M_1) \in P$
  - 存在图灵机M_2使得$L(M_2) \notin P$
- 等价地形式：
  - 存在某个图灵可识别的语言$L_1 \in P$
  - 存在某个图灵可识别的语言$L_2 \notin P$

Rice's Theorem

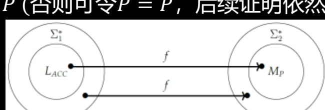
47

定理 **莱斯定理, Rice's Theorem**

令P是图灵可识别语言的任意一个非平凡性质，并且语言$M_P = \{\langle M \rangle | L(M) \in P\}$，那么$M_P$是不可判定的。

证明概要 构造映射归约证明$L_{ACC} \le_m M_P$

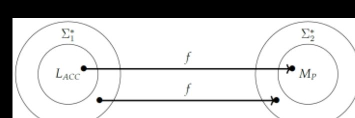
- 假设$\emptyset \notin P$ (否则可令$P = \bar{P}$，后续证明依然有效)



- 令M_1是任一满足$L(M_1) \in P$的图灵机，因此$\langle M_1 \rangle \in M_P$
- 令M_2是满足$L(M_2) = \emptyset$的图灵机，因此$\langle M_2 \rangle \notin M_P$

Rice's Theorem

48



构造$f(x)$

- 如果x不是$\langle M, w \rangle$合法形式，返回$f(x) = \langle M_2 \rangle$
- 如果x形如$\langle M, w \rangle$，返回$f(x) = \langle M' \rangle$
  - M'的输入是y
  - M'模拟M在w上的运行
    - 如果M接受w，在y上模拟M_1运行，若M_1接受y则M'接受
    - 如果M不接受w或者M_1不接受y，一直运行不停机
- $x \in L_{ACC} \Leftrightarrow f(x) \in M_P$

Rice's Theorem

49

☑$\{\langle M\rangle | M$ is a TM that accepts at least 37 different strings$\}$

☒$\{\langle M\rangle | M$ is a TM that has at least 37 states$\}$

☒$\{\langle M\rangle | M$ is a TM that runs for at most 37 steps on input 01$\}$

☒$\{\langle M\rangle | M$ is a TM that accepts the string 01 in exactly an even number of steps$\}$

不适用于Rice定理，但性质是undecidable的，可以用一个从L_{ACC01}的归约证明

☒$\{\langle M\rangle | M$ is a TM and $L(M)$ is recognized by some TM having an even number of states$\}$

平凡性质

Rice's Theorem

50

☑$\{\langle M\rangle | M$ is a TM and $L(M)$ is recognized by some TM having at most 37 states and at most 37 tape symbols$\}$

☒$\{\langle M\rangle | M$ is a TM that has at most 37 states and at most 37 tape symbols$\}$

☒$\{\langle M\rangle | M$ is a TM and $L(M)$ is recognized by some TM having at least 37 states and at least 37 tape symbols$\}$

51

上述不可判定问题看起来**不自然**

波斯特对应问题

52

定义 波斯特对应问题(Post Correspondence Problem)

给定两个对应的序列

$$A = (\alpha_1, \alpha_2, \dots, \alpha_k)$$
$$B = (\beta_1, \beta_2, \dots, \beta_k)$$

其中$\alpha_i \in \Sigma^*$且$\beta_j \in \Sigma^*$，问是否存在一个有限的下标序列

$$\langle i_1, i_2, \dots, i_m\rangle$$

使得

$$\alpha_{i_1}\alpha_{i_2}\dots\alpha_{i_m} = \beta_{i_1}\beta_{i_2}\dots\beta_{i_m}$$

- $A = (1, 10111, 10), B = (111, 10, 0)$
- $A = (10, 011, 101), B = (101, 11, 011)$

波斯特对应问题

53

定义 Modified Post Correspondence Problem (MPCP)

给定两个对应的序列

$$A = (\alpha_1, \alpha_2, \dots, \alpha_k)$$
$$B = (\beta_1, \beta_2, \dots, \beta_k)$$

其中$\alpha_i \in \Sigma^*$且$\beta_j \in \Sigma^*$，问是否存在一个有限的下标序列

$$\langle i_1, i_2, \dots, i_m\rangle$$

使得

$$\alpha_1\alpha_{i_1}\alpha_{i_2}\dots\alpha_{i_m} = \beta_1\beta_{i_1}\beta_{i_2}\dots\beta_{i_m}$$

- $A = (1,10111,10), B = (111,10,0)$

波斯特对应问题

54

定理 修改后的波斯特对应问题是不可判定的.

规则	A	B	含义
(1)	$\#$	$\#q_0w\#$	w为输入，q_0为初始状态
(2)	Z $\#$	Z $\#$	Z取遍所有带上字符
(3)	qX ZqX $q\#$ $Zq\#$	Yp pZY $Yp\#$ $pZY\#$	$\delta(q,X) = (p,Y,R)$ $\delta(q,X) = (p,Y,L)$ $\delta(q,B) = (p,Y,R)$ $\delta(q,B) = (p,Y,L)$
(4)	$Z_1q_fZ_2$ Zq_f q_fZ	q_f q_f q_f	
(5)	$q_f\#\#$	$\#$	

波斯特对应问题

55

定理 PCP
波斯特对应问题是不可判定的.

证明思路 建立从MPCP到PCP的归约

- MPCP的输入： $A = (1, 10111, 10),\ B = (111, 10, 0)$
- PCP对应输入：
  $C\ =\ (*1*, 1*, 1*0*1*1*1*, 1*0*, \$)$
  $D\ =\ (*1*1*1, *1*1*1, *1*0, *0, *\$)$

归约视角 $L_{ACC} \le_m$ MPCP $\le_m$ PCP
归约f: $\langle M, w\rangle \to \langle A_{M,w}, B_{M,w}\rangle$
➤ $\langle M, w\rangle \in L_{ACC}$ 当且仅当$\langle A_{M,w}, B_{M,w}\rangle \in$MPCP
归约g: $\langle A_{M,w}, B_{M,w}\rangle \to \langle C_{M,w}, D_{M,w}\rangle$
➤ $\langle A_{M,w}, B_{M,w}\rangle \in$MPCP当且仅当$\langle C_{M,w}, D_{M,w}\rangle \in$PCP

文法相关的问题

56

$G = (V, \Sigma, R, S)$, $V = \{S, a, b, c, A, B, C, T_a, T_b, T_c\}$, $\Sigma = \{a, b, c\}$, R:

- $S \to ABCS,\ S \to T_c$
- $CA \to AC,\ BA \to AB,\ CB \to BC$
- $CT_c \to T_c c,\ CT_c \to T_b c,\ BT_b \to T_b b,\ BT_b \to T_a b,\ AT_a \to T_a a$
- $T_a \to \epsilon$

不出意外，下面每一个问题都**不可判定**

- 给定文法G和字符串w，判断是否$w \in L(G)$
- 给定文法G，判断是否$\epsilon \in L(G)$
- 给定文法G_1和G_2，判断是否$L(G_1) = L(G_2)$
- 给定文法G，判断是否$L(G) = \emptyset$
- 存在一个特定的文法G_0，判断给定字符串w是否属于语言$L(G_0)$

文法相关的问题

57

上下文无关文法是否更简单？

- 判断是否$w \in L(G)$ **可判定，有算法**
- 判断是否$\epsilon \in L(G)$ **可判定，有算法**
- 判断是否$L(G_1) = L(G_2)$ **不可判定**
- 判断是否$L(G) = \emptyset$ **可判定，有算法**

定理 **上下文无关文法相关问题**
如下问题都是不可判定的。
❶给定上下文无关文法G，是否$L(G) = \Sigma^*$
❷给定上下文无关文法G_1和G_2，是否$L(G_1) = L(G_2)$
❸给定下推自动机M_1和M_2，是否接受相同语言
❹给定下推自动机M，计算一个等价的下推自动机M'，使得M'的状态最少

58

计算复杂性理论