可判定性

吴　铤

2017 年 11 月 13 日

可判定性

1 可判定语言

2 不可判定性

可判定性

- DFA的空性质测试：判断某个DFA是否不接受任何字符串。令$E_{DFA} = \{\langle A\rangle | A$是DFA, 且$L(A) = \emptyset\}$

定理1 (E_{DFA}是可判定语言)

证明.

对于输入$\langle A\rangle$

- 标记A的起始状态；
- 重复以下步骤，直至被标记集合不变：
  - 对于未标记某个状态，如果有一个到达它的转移是从已标记状态出发的，则将其标记；
- 如果没有接受状态被标记，则接受；否则拒绝。

- **DFA的空性质测试**：判断某个DFA是否不接受任何字符串。令$E_{DFA} = \{\langle A \rangle | A$是DFA, 且$L(A) = \emptyset\}$

定理1 (E_{DFA}是可判定语言)

证明.

对于输入$\langle A \rangle$

1. 标记A的起始状态；
2. 重复以下步骤，直至被标记集合不增：
   - 对于未标记某个状态，如果有一个到达它的转移是从已标记状态出发的，则将其标记；
3. 如果没有接受状态被标记，则接受；否则拒绝。

□

- DFA的空性质测试：判断某个DFA是否不接受任何字符串。令$E_{DFA}=\{\langle A\rangle | A$是DFA, 且$L(A)=\emptyset\}$

定理1 (E_{DFA}是可判定语言)

证明.

对于输入$\langle A\rangle$

1. 标记A的起始状态；
2. 重复以下步骤，直至被标记集合不增：
   1. 对于未标记某个状态，如果有一个到达它的转移是从已标记状态出发的，则将其标记；
3. 如果没有接受状态被标记，则接受；否则拒绝。

□

- **DFA的空性质测试**：判断某个DFA是否不接受任何字符串。令$E_{DFA} = \{\langle A\rangle | A\text{是DFA, 且}L(A) = \emptyset\}$

定理1 (E_{DFA}是可判定语言)

证明.

对于输入$\langle A\rangle$

1. 标记A的起始状态；
2. 重复以下步骤，直至被标记集合不增：
   - 对于未标记某个状态，如果有一个到达它的转移是从已标记状态出发的，则将其标记；
3. 如果没有接受状态被标记，则接受；否则拒绝。

∎

- DFA的空性质测试：判断某个DFA是否不接受任何字符串。令$E_{DFA} = \{\langle A\rangle | A$是DFA, 且$L(A) = \emptyset\}$

定理1 (E_{DFA}是可判定语言)

证明.

对于输入$\langle A\rangle$

1. 标记A的起始状态；
2. 重复以下步骤，直至被标记集合不增：
   1. 对于未标记某个状态，如果有一个到达它的转移是从已标记状态出发的，则将其标记；
3. 如果没有接受状态被标记，则接受；否则拒绝。

□

- **DFA的空性质测试**：判断某个DFA是否不接受任何字符串。令$E_{DFA} = \{\langle A\rangle | A是\text{DFA}, 且L(A) = \emptyset\}$

定理1 (E_{DFA}是可判定语言)

证明.

对于输入$\langle A\rangle$

1. 标记A的起始状态；
2. 重复以下步骤，直至被标记集合不增：
   1. 对于未标记某个状态，如果有一个到达它的转移是从已标记状态出发的，则将其标记；
3. 如果没有接受状态被标记，则接受；否则拒绝。

□

- DFA的空性质测试：判断某个DFA是否不接受任何字符串。令$E_{DFA} = \{\langle A \rangle | A\text{是DFA}, \text{且}L(A) = \emptyset\}$

定理1 (E_{DFA}是可判定语言)

证明.

对于输入$\langle A \rangle$

1. 标记A的起始状态；
2. 重复以下步骤，直至被标记集合不增：
   1. 对于未标记某个状态，如果有一个到达它的转移是从已标记状态出发的，则将其标记；
3. 如果没有接受状态被标记，则接受；否则拒绝。

□

- DFA的空性质测试：判断某个DFA是否不接受任何字符串。令$E_{DFA} = \{\langle A\rangle | A$是DFA, 且$L(A) = \emptyset\}$

定理1 (E_{DFA}是可判定语言)

证明.

对于输入$\langle A\rangle$

1. 标记A的起始状态；
2. 重复以下步骤，直至被标记集合不增：
   1. 对于未标记某个状态，如果有一个到达它的转移是从已标记状态出发的，则将其标记；
3. 如果没有接受状态被标记，则接受；否则拒绝。

□

- DFA的空性质测试：判断某个DFA是否不接受任何字符串。令$E_{DFA} = \{\langle A\rangle | A\text{是DFA, 且}L(A) = \emptyset\}$

定理1 (E_{DFA}是可判定语言)

证明.

对于输入$\langle A\rangle$

1. 标记A的起始状态；
2. 重复以下步骤，直至被标记集合不增：
   1. 对于未标记某个状态，如果有一个到达它的转移是从已标记状态出发的，则将其标记；
3. 如果没有接受状态被标记，则接受；否则拒绝。

$\square$

定理2 ($EQ_{DFA} = \{\langle A, B\rangle | A, B$是DFA$, L(A) = L(B)\}$是可判定语言)

证明.

考虑$L(A), L(B)$的对称差:

$$\begin{aligned} L(C) &\stackrel{def}{=} L(A) \oplus L(B) \\ &= (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B)) \end{aligned}$$

则就是要判断$\langle C\rangle \stackrel{?}{\in} E_{DFA}$。 □

定理2 ($EQ_{DFA} = \{\langle A, B\rangle | A, B$是DFA, $L(A) = L(B)\}$是可判定语言)

证明.

考虑$L(A), L(B)$的对称差：

$$L(C) \stackrel{def}{=} L(A) \oplus L(B)$$
$$= (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B))$$

则就是要判断$\langle C\rangle \stackrel{?}{\in} E_{DFA}$。 □

定理2 ($EQ_{DFA} = \{\langle A, B\rangle | A, B$是DFA, $L(A) = L(B)\}$是可判定语言)

证明.

考虑$L(A), L(B)$的对称差：

$$
\begin{aligned}
L(C) &\stackrel{def}{=} L(A) \oplus L(B) \\
&= (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B))
\end{aligned}
$$

则就是要判断$\langle C\rangle \stackrel{?}{\in} E_{DFA}$。 □

定理2 ($EQ_{DFA} = \{\langle A, B\rangle | A, B$是DFA$, L(A) = L(B)\}$是可判定语言)

证明.

考虑$L(A), L(B)$的对称差：

$$L(C) \overset{def}{=} L(A) \oplus L(B)$$
$$= (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B))$$

则就是要判断$\langle C\rangle \overset{?}{\in} E_{DFA}$。 $\square$

- DFA接受问题：设B, w分别为给定的DFA和串，判断B是否接受w。
  - 构造图灵机M如下
    - 检查输入的$\langle B, w\rangle$；
    - 在图灵机M上模拟B在输入为w上的运行；
    - 如果最终B进入接受状态，则接受，否则拒绝；
- NFA接受问题：设B, w分别为给定的NFA和串，判断B是否接受w。
- 正则表达式的派生问题：设R, w分别为给定的正则表达式和串，判断R是否派生w。

- DFA接受问题：设B, w分别为给定的DFA和串，判断B是否接受w。
  - 构造图灵机M如下
    1. 检查输入的$\langle B, w\rangle$；
    2. 在图灵机M上模拟B在输入为w上的运行；
    3. 如果最终B进入接受状态，则接受，否则拒绝；
- NFA接受问题：设B, w分别为给定的NFA和串，判断B是否接受w。
- 正则表达式的派生问题：设R, w分别为给定的正则表达式和串，判断R是否派生w。

- DFA接受问题：设B, w分别为给定的DFA和串，判断B是否接受w。
  - 构造图灵机M如下
    1. 检查输入的$\langle B, w\rangle$；
    2. 在图灵机M上模拟B在输入为w上的运行；
    3. 如果最终B进入接受状态，则接受，否则拒绝；
- NFA接受问题：设B, w分别为给定的NFA和串，判断B是否接受w。
- 正则表达式的派生问题：设R, w分别为给定的正则表达式和串，判断R是否派生w。

- DFA接受问题：设B, w分别为给定的DFA和串，判断B是否接受w。
  - 构造图灵机M如下
    1. 检查输入的$\langle B, w\rangle$;
    2. 在图灵机M上模拟B在输入为w上的运行;
    3. 如果最终B进入接受状态，则接受，否则拒绝;
- NFA接受问题：设B, w分别为给定的NFA和串，判断B是否接受w。
- 正则表达式的派生问题：设R, w分别为给定的正则表达式和串，判断R是否派生w。

- DFA接受问题：设B, w分别为给定的DFA和串，判断B是否接受w。
  - 构造图灵机M如下
    1. 检查输入的$\langle B, w \rangle$；
    2. 在图灵机M上模拟B在输入为w上的运行；
    3. 如果最终B进入接受状态，则接受，否则拒绝；
- NFA接受问题：设B, w分别为给定的NFA和串，判断B是否接受w。
- 正则表达式的派生问题：设R, w分别为给定的正则表达式和串，判断R是否派生w。

- DFA接受问题：设B, w分别为给定的DFA和串，判断B是否接受w。
  - 构造图灵机M如下
    1. 检查输入的$\langle B, w\rangle$；
    2. 在图灵机M上模拟B在输入为w上的运行；
    3. 如果最终B进入接受状态，则接受，否则拒绝；
- NFA接受问题：设B, w分别为给定的NFA和串，判断B是否接受w。
- 正则表达式的派生问题：设R, w分别为给定的正则表达式和串，判断R是否派生w。

- DFA接受问题：设B, w分别为给定的DFA和串，判断B是否接受w。
  - 构造图灵机M如下
    1. 检查输入的$\langle B, w \rangle$；
    2. 在图灵机M上模拟B在输入为w上的运行；
    3. 如果最终B进入接受状态，则接受，否则拒绝；
- NFA接受问题：设B, w分别为给定的NFA和串，判断B是否接受w。
- 正则表达式的派生问题：设R, w分别为给定的正则表达式和串，判断R是否派生w。

- DFA接受问题：设B, w分别为给定的DFA和串，判断B是否接受w。
  - 构造图灵机M如下
    1. 检查输入的$\langle B, w\rangle$；
    2. 在图灵机M上模拟B在输入为w上的运行；
    3. 如果最终B进入接受状态，则接受，否则拒绝；
- NFA接受问题：设B, w分别为给定的NFA和串，判断B是否接受w。
- 正则表达式的派生问题：设R, w分别为给定的正则表达式和串，判断R是否派生w。

$$A_{CFG} = \{\langle G, w\rangle | G\text{是}CFG, G\text{派生串}w\}$$

$$E_{CFG} = \{\langle G\rangle | G\text{是}CFG, L(G) = \emptyset\}$$

$$EQ_{CFG} = \{\langle G, H\rangle | G, H\text{都是}CFG, L(G) = L(H)\}$$

定理3 (A_{CFG}是可判定语言)

证明：按如下方式构造TM S：

对于输入$\langle G, w\rangle$，

1. 将G转换为与之等价的乔姆斯基范式；
2. 列出所有$2n-1$步的派生，其中n是w的长度；
3. 如果其中有一个产生w，则接受，否则拒绝。

定理3 (A_{CFG}是可判定语言)

证明：按如下方式构造TM S：

对于输入$\langle G, w\rangle$，

1. 将G转换为与之等价的乔姆斯基范式；
2. 列出所有$2n-1$步的派生，其中n是w的长度；
3. 如果其中有一个产生w，则接受，否则拒绝。

定理3 (A_{CFG}是可判定语言)

证明：按如下方式构造TM S：
对于输入$\langle G, w\rangle$，

1. 将G转换为与之等价的乔姆斯基范式；
2. 列出所有$2n-1$步的派生，其中n是w的长度；
3. 如果其中有一个产生w，则接受，否则拒绝。

定理3 (A_{CFG}是可判定语言)

证明：按如下方式构造TM S：
对于输入$\langle G, w\rangle$，

1. 将G转换为与之等价的乔姆斯基范式；
2. 列出所有$2n-1$步的派生，其中n是w的长度；
3. 如果其中有一个产生w，则接受，否则拒绝。

定理3 (A_{CFG}是可判定语言)

证明：按如下方式构造TM S：

对于输入$\langle G, w\rangle$，

1. 将G转换为与之等价的乔姆斯基范式；
2. 列出所有$2n-1$步的派生，其中n是w的长度；
3. 如果其中有一个产生w，则接受，否则拒绝。

定理3 (A_{CFG}是可判定语言)

证明：按如下方式构造TM S：
对于输入$\langle G, w\rangle$，

1. 将G转换为与之等价的乔姆斯基范式；
2. 列出所有$2n-1$步的派生，其中n是w的长度；
3. 如果其中有一个产生w，则接受，否则拒绝。

定理4 (E_{CFG}是可判定语言)

证明.

对于输入的CFG$\langle G\rangle$:

1. 将G中所有的终结符全都作上标记;
2. 重复以下步骤，直至找不到可以标记的变元:
   - 若G有规则$A \to U_1U_2\cdots U_k$，且$U_i, i=1,\cdots,k$中的每个符号都已标记，则标记变元A;
3. 如果起始状态没有被标记，则接受，否则拒绝。

定理5 (EQ_{CFG}不是可判定的)

定理4 (E_{CFG}是可判定语言)

证明.

对于输入的CFG$\langle G \rangle$：

1. 将G中所有的终结符全都作上标记；
2. 重复以下步骤，直至找不到可以标记的变元：
   1. 若G有规则$A \to U_1 U_2 \cdots U_k$，且$U_i, i = 1, \cdots, k$中的每个符号都已标记，则标记变元A；
3. 如果起始状态没有被标记，则接受，否则拒绝。

定理5 (EQ_{CFG}不是可判定的)

定理4 (E_{CFG}是可判定语言)

证明.

对于输入的CFG$\langle G\rangle$:

1. 将G中所有的终结符全都作上标记;
2. 重复以下步骤，直至找不到可以标记的变元:
    1. 若G有规则$A \to U_1U_2\cdots U_k$，且$U_i, i=1,\cdots,k$中的每个符号都已标记，则标记变元A;
3. 如果起始状态没有被标记，则接受，否则拒绝。

定理5 (EQ_{CFG}不是可判定的)

定理4 (E_{CFG}是可判定语言)

证明.

对于输入的CFG$\langle G\rangle$：

1. 将G中所有的终结符全都作上标记；
2. 重复以下步骤，直至找不到可以标记的变元：
   1. 若G有规则$A \to U_1U_2\cdots U_k$，且$U_i, i=1,\cdots,k$中的每个符号都已标记，则标记变元A；
3. 如果起始状态没有被标记，则接受，否则拒绝。

定理5 (EQ_{CFG}不是可判定的)

定理4 (E_{CFG}是可判定语言)

证明.

对于输入的CFG$\langle G\rangle$：

1. 将G中所有的终结符全都作上标记；
2. 重复以下步骤，直至找不到可以标记的变元：
   1. 若G有规则$A \to U_1U_2\cdots U_k$，且$U_i, i = 1, \cdots, k$中的每个符号都已标记，则标记变元A；
3. 如果起始状态没有被标记，则接受，否则拒绝。

定理5 (EQ_{CFG}不是可判定的)

定理4 (E_{CFG}是可判定语言)

证明.

对于输入的CFG$\langle G\rangle$：

1. 将G中所有的终结符全都作上标记；
2. 重复以下步骤，直至找不到可以标记的变元：
   1. 若G有规则$A \rightarrow U_1U_2\cdots U_k$，且$U_i, i=1,\cdots,k$中的每个符号都已标记，则标记变元$A$；
3. 如果起始状态没有被标记，则接受，否则拒绝。

定理5 (EQ_{CFG}不是可判定的)

定理4 (E_{CFG}是可判定语言)

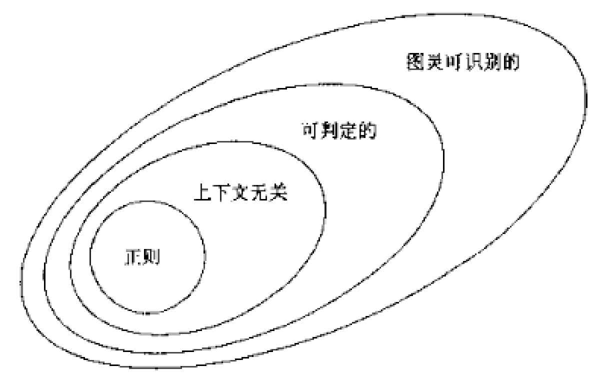
证明.

对于输入的CFG$\langle G\rangle$:

1. 将G中所有的终结符全都作上标记;
2. 重复以下步骤，直至找不到可以标记的变元:
   1. 若G有规则$A \rightarrow U_1U_2\cdots U_k$，且$U_i, i=1,\cdots,k$中的每个符号都已标记，则标记变元$A$;
3. 如果起始状态没有被标记，则接受，否则拒绝。

定理5 (EQ_{CFG}不是可判定的)

与上下文无关语言相关的可判定型问题



停机问题：$A_{TM} = \{\langle M, w\rangle | M$是一个$TM$, 且接受$w\}$

定理6 (A_{TM}是可识别不可判定的)

可识别的证明.

设置通用图灵机U，对于输入$\langle M, w\rangle$执行：

1. 对于输入w，模拟M；
2. 若M进入接受状态，则接受；
   如果进入拒绝状态，则拒绝。

□

停机问题：$A_{TM} = \{\langle M, w\rangle | M\text{是一个}TM, \text{且接受}w\}$

定理6 (A_{TM}是可识别不可判定的)

可识别的证明.

设置通用图灵机U，对于输入$\langle M, w\rangle$执行：

1. 对于输入w，模拟M；
2. 若M进入接受状态，则接受；
   如果进入拒绝状态，则拒绝。

停机问题：$A_{TM} = \{\langle M, w\rangle | M$是一个$TM$, 且接受$w\}$

定理6 (A_{TM}是可识别不可判定的)

可识别的证明.

设置通用图灵机U，对于输入$\langle M, w\rangle$执行：

1. 对于输入w，模拟M；
2. 若M进入接受状态，则接受；
   如果进入拒绝状态，则拒绝。

□

- 对于任意字母表Σ，其上所有串构成的集合Σ^*必定可数；
- 由所有图灵机构成的集合是可数的；
- 字母表Σ上所有语言构成的集合$\mathcal{L}$是不可数的。

定理7

存在不能被任何图灵机识别的语言。

- 对于任意字母表$\sum$，其上所有串构成的集合$\sum^*$必定可数；
- 由所有图灵机构成的集合是可数的；
- 字母表$\sum$上所有语言构成的集合$\mathcal{L}$是不可数的。

定理7

存在不能被任何图灵机识别的语言。

- 对于任意字母表$\sum$，其上所有串构成的集合$\sum^*$必定可数；
- 由所有图灵机构成的集合是可数的；
- 字母表$\sum$上所有语言构成的集合$\mathcal{L}$是不可数的。

定理7

存在不能被任何图灵机识别的语言。

- 对于任意字母表$\sum$，其上所有串构成的集合$\sum^*$必定可数；
- 由所有图灵机构成的集合是可数的；
- 字母表$\sum$上所有语言构成的集合$\mathcal{L}$是不可数的。

定理7

存在不能被任何图灵机识别的语言。

- 对于任意字母表$\sum$，其上所有串构成的集合$\sum^*$必定可数；
- 由所有图灵机构成的集合是可数的；
- 字母表$\sum$上所有语言构成的集合$\mathcal{L}$是不可数的。

定理7

存在不能被任何图灵机识别的语言。

不可判定性的证明.

假设A_{TM}是可判定的，H为其判定器。即对于输入$\langle M, w\rangle$

- 若M接受w，则H接受$\langle M, w\rangle$；
- 若M不接受w，则H拒绝$\langle M, w\rangle$；

即

$$H(\langle M, w\rangle) = \begin{cases} \text{接受} & \text{若} M \text{接受} w \\ \text{拒绝} & \text{若} M \text{不接受} w \end{cases}$$

□

不可判定性的证明.

假设A_{TM}是可判定的，H为其判定器。即对于输入$\langle M, w\rangle$

- 若M接受w，则H接受$\langle M, w\rangle$；
- 若M不接受w，则H拒绝$\langle M, w\rangle$；

即

$$H(\langle M, w\rangle) = \begin{cases} \text{接受} & \text{若}M\text{接受}w \\ \text{拒绝} & \text{若}M\text{不接受}w \end{cases}$$

$\square$

不可判定性的证明.

假设A_{TM}是可判定的，H为其判定器。即对于输入$\langle M, w\rangle$

- 若M接受w，则H接受$\langle M, w\rangle$；
- 若M不接受w，则H拒绝$\langle M, w\rangle$；

即

$$H(\langle M, w\rangle) = \begin{cases} \text{接受} & \text{若}M\text{接受}w \\ \text{拒绝} & \text{若}M\text{不接受}w \end{cases}$$

$\square$

不可判定性的证明.

假设A_{TM}是可判定的，H为其判定器。即对于输入$\langle M, w\rangle$

- 若M接受w，则H接受$\langle M, w\rangle$；
- 若M不接受w，则H拒绝$\langle M, w\rangle$；

即

$$H(\langle M, w\rangle) = \begin{cases} \text{接受} & \text{若}M\text{接受}w \\ \text{拒绝} & \text{若}M\text{不接受}w \end{cases}$$

$\square$

不可判定性的证明.

假设A_{TM}是可判定的，H为其判定器。即对于输入$\langle M, w\rangle$

- 若M接受w，则H接受$\langle M, w\rangle$；
- 若M不接受w，则H拒绝$\langle M, w\rangle$；

即

$$H(\langle M, w\rangle) = \begin{cases} \text{接受} & \text{若}M\text{接受}w \\ \text{拒绝} & \text{若}M\text{不接受}w \end{cases}$$

□

图灵机D的构造：对于输入的对图灵机M的描述$\langle M\rangle$，执行

- 若M接受$\langle M\rangle$，则拒绝；
- 若M不接受$\langle M\rangle$，则接受；

即

$$D(\langle M\rangle)=\begin{cases}\text{接受} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{拒绝}\\ \text{拒绝} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{接受}\end{cases}$$

考虑$D(\langle D\rangle)$：

$$D(\langle D\rangle)=\begin{cases}\text{接受} & \text{若}D\text{不接受}\langle D\rangle\\ \text{拒绝} & \text{若}D\text{接受}\langle D\rangle\end{cases}$$

图灵机D的构造：对于输入的对图灵机M的描述$\langle M\rangle$，执行

- 若M接受$\langle M\rangle$，则拒绝；
- 若M不接受$\langle M\rangle$，则接受；

即

$$D(\langle M\rangle)=\begin{cases}\text{接受} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{拒绝}\\ \text{拒绝} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{接受}\end{cases}$$

考虑$D(\langle D\rangle)$：

$$D(\langle D\rangle)=\begin{cases}\text{接受} & \text{若}D\text{不接受}\langle D\rangle\\ \text{拒绝} & \text{若}D\text{接受}\langle D\rangle\end{cases}$$

图灵机D的构造：对于输入的对图灵机M的描述$\langle M\rangle$，执行

- 若M接受$\langle M\rangle$，则拒绝；
- 若M不接受$\langle M\rangle$，则接受；

即

$$D(\langle M\rangle)=\begin{cases}\text{接受} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{拒绝}\\ \text{拒绝} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{接受}\end{cases}$$

考虑$D(\langle D\rangle)$：

$$D(\langle D\rangle)=\begin{cases}\text{接受} & \text{若}D\text{不接受}\langle D\rangle\\ \text{拒绝} & \text{若}D\text{接受}\langle D\rangle\end{cases}$$

图灵机D的构造：对于输入的对图灵机M的描述$\langle M\rangle$，执行

- 若M接受$\langle M\rangle$，则拒绝；
- 若M不接受$\langle M\rangle$，则接受；

即

$$D(\langle M\rangle)=\begin{cases}\text{接受} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{拒绝}\\ \text{拒绝} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{接受}\end{cases}$$

考虑$D(\langle D\rangle)$：

$$D(\langle D\rangle)=\begin{cases}\text{接受} & \text{若}D\text{不接受}\langle D\rangle\\ \text{拒绝} & \text{若}D\text{接受}\langle D\rangle\end{cases}$$

图灵机D的构造：对于输入的对图灵机M的描述$\langle M\rangle$，执行

- 若M接受$\langle M\rangle$，则拒绝；
- 若M不接受$\langle M\rangle$，则接受；

即

$$D(\langle M\rangle)=\begin{cases}\text{接受} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{拒绝}\\ \text{拒绝} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{接受}\end{cases}$$

考虑$D(\langle D\rangle)$：

$$D(\langle D\rangle)=\begin{cases}\text{接受} & \text{若}D\text{不接受}\langle D\rangle\\ \text{拒绝} & \text{若}D\text{接受}\langle D\rangle\end{cases}$$

图灵机D的构造：对于输入的对图灵机M的描述$\langle M\rangle$，执行

- 若M接受$\langle M\rangle$，则拒绝；
- 若M不接受$\langle M\rangle$，则接受；

即

$$D(\langle M\rangle)=\begin{cases}\text{接受} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{拒绝}\\ \text{拒绝} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{接受}\end{cases}$$

考虑$D(\langle D\rangle)$：

$$D(\langle D\rangle)=\begin{cases}\text{接受} & \text{若}D\text{不接受}\langle D\rangle\\ \text{拒绝} & \text{若}D\text{接受}\langle D\rangle\end{cases}$$

图灵机D的构造：对于输入的对图灵机M的描述$\langle M\rangle$，执行

- 若M接受$\langle M\rangle$，则拒绝；
- 若M不接受$\langle M\rangle$，则接受；

即

$$D(\langle M\rangle)=\begin{cases}\text{接受} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{拒绝}\\ \text{拒绝} & \text{若}H(\langle M,\langle M\rangle\rangle)=\text{接受}\end{cases}$$

考虑$D(\langle D\rangle)$：

$$D(\langle D\rangle)=\begin{cases}\text{接受} & \text{若}D\text{不接受}\langle D\rangle\\ \text{拒绝} & \text{若}D\text{接受}\langle D\rangle\end{cases}$$

定义1 (补图灵可识别)

若某个语言是一个图灵可识别语言的补集，则称其是补图灵可识别的。

定理8

某个语言可判定的充要条件是：其即是图灵可识别的，又是补图灵可识别的。

定义1 (补图灵可识别)

若某个语言是一个图灵可识别语言的补集，则称其是**补图灵可识别的**。

定理8

某个语言可判定的充要条件是：其即是图灵可识别的，又是补图灵可识别的。

定义1 (补图灵可识别)

若某个语言是一个图灵可识别语言的补集，则称其是**补图灵可识别的**。

定理8

某个语言可判定的充要条件是：其即是图灵可识别的，又是补图灵可识别的。

“$\Leftarrow$” 证明.

若$A, \overline{A}$都是图灵可识别的，M_1, M_2分别是$A, \overline{A}$的识别器，则

图灵机M的构造：对于输入w

1. 对输入w并行地运行M_1, M_2;
2. 如果M_1接受，则接受;
   如果M_2接受，则拒绝

$\square$

定理9 ($\overline{A_{TM}}$不是图灵可识别的)

“$\Leftarrow$” 证明.

若$A, \overline{A}$都是图灵可识别的，M_1, M_2分别是$A, \overline{A}$的识别器，则

图灵机M的构造：对于输入w

1. 对输入w并行地运行M_1, M_2;
2. 如果M_1接受，则接受；
   如果M_2接受，则拒绝

$\square$

定理9 ($\overline{A_{TM}}$不是图灵可识别的)

“$\Leftarrow$” 证明.

若$A, \overline{A}$都是图灵可识别的，M_1, M_2分别是$A, \overline{A}$的识别器，则

图灵机M的构造：对于输入w

1. 对输入w并行地运行M_1, M_2;
2. 如果M_1接受，则接受;
   如果M_2接受，则拒绝

□

定理9 ($\overline{A_{TM}}$不是图灵可识别的)

“$\Leftarrow$” 证明.

若$A, \overline{A}$都是图灵可识别的，M_1, M_2分别是$A, \overline{A}$的识别器，则

图灵机M的构造：对于输入w

1. 对输入w并行地运行M_1, M_2；
2. 如果M_1接受，则接受；
   如果M_2接受，则拒绝

□

定理9 ($\overline{A_{TM}}$不是图灵可识别的)

“$\Leftarrow$” 证明.

若$A, \overline{A}$都是图灵可识别的，M_1, M_2分别是$A, \overline{A}$的识别器，则

图灵机M的构造：对于输入w

1. 对输入w并行地运行M_1, M_2；
2. 如果M_1接受，则接受；
   如果M_2接受，则拒绝

□

定理9 ($\overline{A_{TM}}$不是图灵可识别的)

"$\Leftarrow$" 证明.

若$A, \overline{A}$都是图灵可识别的，M_1, M_2分别是$A, \overline{A}$的识别器，则

图灵机M的构造：对于输入w

1. 对输入w并行地运行M_1, M_2；
2. 如果M_1接受，则接受；
   如果M_2接受，则拒绝

$\square$

定理9 ($\overline{A_{TM}}$不是图灵可识别的)

$$HALT_{TM} = \{\langle M, w\rangle | M\text{是一个}TM, \text{且对输入}w\text{停机}\}$$

$$E_{TM} = \{\langle M\rangle | M\text{是一个}TM, L(M) = \emptyset\}$$

$$REGULAR_{TM} = \{\langle M\rangle | M\text{是一个}TM, L(M)\text{是正则语言}\}$$

$$EQ_{TM}\{\langle M_1, M_2\rangle | M_1, M_2\text{是}TM, L(M_1) = L(M_2)\}$$

定理10 ($HALT_{TM}$是不可判定的)

证明：假设TM R判定$HALT_{TM}$，则构造TM S如下：
对于输入$\langle M, w\rangle$，S执行

1. 对于输入$\langle M, w\rangle$执行TM R；
2. 如果R拒绝，则拒绝；否则
   1. 在w上模拟M，直至其停机
   2. 如果M接受，则接受，否则拒绝。

则TM S是判定A_{TM}的图灵机，矛盾。

定理10 ($HALT_{TM}$是不可判定的)

证明：假设TM R判定$HALT_{TM}$，则构造TM S如下：

对于输入$\langle M, w\rangle$，S执行

1. 对于输入$\langle M, w\rangle$执行TM R;
2. 如果R拒绝，则拒绝；否则
   1. 在w上模拟M，直至其停机
   2. 如果M接受，则接受，否则拒绝。

则TM S是判定A_{TM}的图灵机，矛盾。

定理10 ($HALT_{TM}$是不可判定的)

证明：假设TM R判定$HALT_{TM}$，则构造TM S如下：

对于输入$\langle M, w\rangle$，S执行

1. 对于输入$\langle M, w\rangle$执行TM R；
2. 如果R拒绝，则拒绝；否则
   1. 在w上模拟M，直至其停机
   2. 如果M接受，则接受，否则拒绝。

则TM S是判定A_{TM}的图灵机，矛盾。

定理10 ($HALT_{TM}$是不可判定的)

证明：假设TM R判定$HALT_{TM}$，则构造TM S如下：对于输入$\langle M, w\rangle$，S执行

1. 对于输入$\langle M, w\rangle$执行TM R；
2. 如果R拒绝，则拒绝；否则
   1. 在w上模拟M，直至其停机
   2. 如果M接受，则接受，否则拒绝。

则TM S是判定A_{TM}的图灵机，矛盾。

定理10 ($HALT_{TM}$是不可判定的)

证明：假设TM R判定$HALT_{TM}$，则构造TM S如下：
对于输入$\langle M, w\rangle$，S执行

1. 对于输入$\langle M, w\rangle$执行TM R;
2. 如果R拒绝，则拒绝；否则
   1. 在w上模拟M，直至其停机
   2. 如果M接受，则接受，否则拒绝。

则TM S是判定A_{TM}的图灵机，矛盾。

定理10 ($HALT_{TM}$是不可判定的)

证明：假设TM R判定$HALT_{TM}$，则构造TM S如下：

对于输入$\langle M, w\rangle$，S执行

1. 对于输入$\langle M, w\rangle$执行TM R；
2. 如果R拒绝，则拒绝；否则
   1. 在w上模拟M，直至其停机
   2. 如果M接受，则接受，否则拒绝。

则TM S是判定A_{TM}的图灵机，矛盾。

定理10 ($HALT_{TM}$是不可判定的)

证明：假设TM R判定$HALT_{TM}$，则构造TM S如下：
对于输入$\langle M, w\rangle$，S执行

1. 对于输入$\langle M, w\rangle$执行TM R；
2. 如果R拒绝，则拒绝；否则
   1. 在w上模拟M，直至其停机
   2. 如果M接受，则接受，否则拒绝。

则TM S是判定A_{TM}的图灵机，矛盾。

定理10 ($HALT_{TM}$是不可判定的)

证明：假设TM R判定$HALT_{TM}$，则构造TM S如下：
对于输入$\langle M, w\rangle$，S执行

1. 对于输入$\langle M, w\rangle$执行TM R;
2. 如果R拒绝，则拒绝；否则
   1. 在w上模拟M，直至其停机
   2. 如果M接受，则接受，否则拒绝。

则TM S是判定A_{TM}的图灵机，矛盾。

定理11 (E_{TM}是不可判定的)

证明:假设TM R判定E_{TM}，则构造TM S如下：

对于输入$\langle M, w\rangle$，执行

1. 构建TM M_1如下：
   - 如果输入$x \neq w$则拒绝，否则在x上运行M，当M接受时，接受；
2. 在$\langle M_1\rangle$上运行R:
   - 如果R接受，则拒绝；如果R拒绝，则接受。

定理11 (E_{TM}是不可判定的)

证明:假设TM R判定E_{TM}，则构造TM S如下：

对于输入$\langle M, w\rangle$，执行

1. 构建TM M_1如下：
   - 如果输入$x \neq w$则拒绝，否则在x上运行M，当M接受时，接受；
2. 在$\langle M_1\rangle$上运行R:
   - 如果R接受，则拒绝；如果R拒绝，则接受。

定理11 (E_{TM}是不可判定的)

证明:假设TM R判定E_{TM}，则构造TM S如下：
对于输入$\langle M, w\rangle$，执行

1. 构建TM M_1如下：
   - 如果输入$x \neq w$则拒绝，否则在x上运行M，当M接受时，接受；
2. 在$\langle M_1\rangle$上运行R:
   - 如果R接受，则拒绝；如果R拒绝，则接受。

定理11 (E_{TM}是不可判定的)

证明:假设TM R判定E_{TM}，则构造TM S如下：
对于输入$\langle M, w\rangle$，执行

1. 构建TM M_1如下：
   - 如果输入$x \neq w$则拒绝，否则在x上运行M，当M接受时，接受；
2. 在$\langle M_1\rangle$上运行R:
   - 如果R接受，则拒绝；如果R拒绝，则接受。

定理11 (E_{TM}是不可判定的)

证明:假设TM R判定E_{TM}，则构造TM S如下：

对于输入$\langle M, w\rangle$，执行

1. 构建TM M_1如下：
   - 如果输入$x \neq w$则拒绝，否则在x上运行M，当M接受时，接受；
2. 在$\langle M_1\rangle$上运行R:
   - 如果R接受，则拒绝；如果R拒绝，则接受。

定理11 (E_{TM}是不可判定的)

证明:假设TM R判定E_{TM}，则构造TM S如下：

对于输入$\langle M, w\rangle$，执行

1. 构建TM M_1如下：
   - 如果输入$x \neq w$则拒绝，否则在x上运行M，当M接受时，接受；
2. 在$\langle M_1\rangle$上运行R:
   - 如果R接受，则拒绝；如果R拒绝，则接受。

定理11 (E_{TM}是不可判定的)

证明:假设TM R判定E_{TM}，则构造TM S如下：

对于输入$\langle M, w\rangle$，执行

1. 构建TM M_1如下：
   - 如果输入$x \neq w$则拒绝，否则在x上运行M，当M接受时，接受；
2. 在$\langle M_1\rangle$上运行R:
   - 如果R接受，则拒绝；如果R拒绝，则接受。