可归约性

吴 铤

2003 年 12 月 18 日

可归约性

1 语言理论中的不可判定问题

可归约性

1 语言理论中的不可判定问题

2 利用计算历史的归约

可归约性

1 语言理论中的不可判定问题

2 利用计算历史的归约

3 映射可归约性

$$A_{TM} = \{\langle M, \omega\rangle \mid M\text{是一个图灵机，且}M\text{接受}\omega\}$$

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

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

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

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

$$A_{TM} = \{\langle M, \omega\rangle \mid M\text{是一个图灵机，且}M\text{接受}\omega\}$$

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

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

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

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

归约：

$$A_{TM} = \{\langle M, \omega\rangle \mid M\text{是一个图灵机，且}M\text{接受}\omega\}$$

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

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

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

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

归约：将问题A“转化”为问题B，使得可以用问题B的解来解决问题A。

$$A_{TM} = \{\langle M, \omega\rangle \mid M\text{是一个图灵机，且}M\text{接受}\omega\}$$

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

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

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

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

归约：将问题A“转化”为问题B，使得可以用问题B的解来解决问题A。

- 如果B是可判定的，那么A也是可判定的；

$$A_{TM} = \{\langle M, \omega\rangle \mid M\text{是一个图灵机，且}M\text{接受}\omega\}$$

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

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

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

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

归约：将问题A“转化”为问题B，使得可以用问题B的解来解决问题A。

- 如果B是可判定的，那么A也是可判定的；
- 如果A是不可判定的，那么B也是不可判定的。

定理1 ($HALT_{TM}$是不可判定的)

定理1 ($HALT_{TM}$是不可判定的)

证明：假设TM R判定$HALT_{TM}$，则构造TM S如下：

定理1 ($HALT_{TM}$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

定理1 ($HALT_{TM}$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;

定理1 ($HALT_{TM}$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

定理1 ($HALT_{TM}$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

定理1 ($HALT_{TM}$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。

定理1 ($HALT_{TM}$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2 (E_{TM}是不可判定的)

定理2 (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：

定理2 (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

定理2 (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：

定理2 (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；

定理2 (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:

定理2 (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3 ($REGULAR_{TM}$是不可判定的)

思路.

$\square$

定理3 ($REGULAR_{TM}$是不可判定的)

思路.

假设R是判定$REGULAR_{TM}$的TM,

$\square$

定理3 ($REGULAR_{TM}$是不可判定的)

思路.

假设R是判定$REGULAR_{TM}$的TM，需构造判定A_{TM}的TM S，

$\square$

定理3 ($REGULAR_{TM}$是不可判定的)

思路.

假设R是判定$REGULAR_{TM}$的TM，需构造判定A_{TM}的TM S，使得对于输入的$\langle M, w\rangle$，S能够判定M是否接受w.

$\square$

定理3 ($REGULAR_{TM}$是不可判定的)

思路.

假设R是判定$REGULAR_{TM}$的TM，需构造判定A_{TM}的TM S，使得对于输入的$\langle M, w\rangle$，S能够判定M是否接受w.

由M设计新TM M_1，使得“M接受$w \Longleftrightarrow M_1$识别正则语言”。 $\square$

M_1的构造：

M_1的构造：

- 当M不接受w时，其识别非正则语言$\{0^n1^n|n \geq 0\}$;

M_1的构造：

- 当M不接受w时，其识别非正则语言$\{0^n1^n|n \geq 0\}$;
- 当M接受w时，其识别正则语言$\sum^*$.

M_1的构造：

- 当M不接受w时，其识别非正则语言$\{0^n1^n|n \geq 0\}$;
- 当M接受w时，其识别正则语言$\sum^*$.

对于输入x，M_1执行：

M_1的构造：

- 当M不接受w时，其识别非正则语言$\{0^n1^n|n \geq 0\}$;
- 当M接受w时，其识别正则语言$\sum^*$.

对于输入x，M_1执行：

1. 如果$x \in \{0^n1^n|n \geq 0\}$，则接受；

M_1的构造：

- 当M不接受w时，其识别非正则语言$\{0^n1^n|n \geq 0\}$;
- 当M接受w时，其识别正则语言$\sum^*$.

对于输入x，M_1执行：

1. 如果$x \in \{0^n1^n|n \geq 0\}$，则接受；
2. 如果$x \notin \{0^n1^n|n \geq 0\}$，则在输入$w$上运行$M$：

M_1的构造：

- 当M不接受w时，其识别非正则语言$\{0^n1^n|n \geq 0\}$;
- 当M接受w时，其识别正则语言$\sum^*$.

对于输入x，M_1执行：

1. 如果$x \in \{0^n1^n|n \geq 0\}$，则接受；
2. 如果$x \notin \{0^n1^n|n \geq 0\}$，则在输入$w$上运行$M$：
   - 如果M接受w，则接受输入x。

M_1的构造：

- 当M不接受w时，其识别非正则语言$\{0^n1^n|n \geq 0\}$;
- 当M接受w时，其识别正则语言$\sum^*$.

对于输入x，M_1执行：

1. 如果$x \in \{0^n1^n|n \geq 0\}$，则接受;
2. 如果$x \notin \{0^n1^n|n \geq 0\}$，则在输入$w$上运行$M$：
   - 如果M接受w，则接受输入x。

$$L(M_1) = \begin{cases} \{0^n1^n|n \geq 0\} & M\text{不接受}w \\ \sum^* & M\text{接受}w \end{cases}$$

假设R是判定$REGULAR_{TM}$的TM，构造判定A_{TM}的TM S：

假设R是判定$REGULAR_{TM}$的TM，构造判定A_{TM}的TM S：

S对于输入的$\langle M, w\rangle$，执行

假设R是判定$REGULAR_{TM}$的TM，构造判定A_{TM}的TM S：

S对于输入的$\langle M, w\rangle$，执行

1. 构造TM M_1;

假设R是判定$REGULAR_{TM}$的TM，构造判定A_{TM}的TM S：

S对于输入的$\langle M, w\rangle$，执行

1. 构造TM M_1;
2. 对于输入$\langle M_1\rangle$，运行R;

假设R是判定$REGULAR_{TM}$的TM，构造判定A_{TM}的TM S：

S对于输入的$\langle M, w\rangle$，执行

1. 构造TM M_1;
2. 对于输入$\langle M_1\rangle$，运行R;
3. 如果R接受，则接受；如果R拒绝，则拒绝。

定理4 (EQ_{TM}是不可判定的)

定理4 (EQ_{TM}是不可判定的)

- 思路：当$L(M_1)=\emptyset$时，E_{TM}是EQ_{TM}的特例。

定理4 (EQ_{TM}是不可判定的)

- 思路：当$L(M_1) = \emptyset$时，E_{TM}是EQ_{TM}的特例。
- 设TM R判定EQ_{TM}，构造判定E_{TM}的TM S如下：

定理4 (EQ_{TM}是不可判定的)

- 思路：当$L(M_1) = \emptyset$时，E_{TM}是EQ_{TM}的特例。
- 设TM R判定EQ_{TM}，构造判定E_{TM}的TM S如下：对于输入$\langle M \rangle$，S执行

定理4 (EQ_{TM}是不可判定的)

- 思路：当$L(M_1) = \emptyset$时，E_{TM}是EQ_{TM}的特例。
- 设TM R判定EQ_{TM}，构造判定E_{TM}的TM S如下：对于输入$\langle M \rangle$，S执行
  1. 在输入$\langle M, M_1 \rangle$上运行R，其中M_1是拒绝所有输入的TM；

定理4 (EQ_{TM}是不可判定的)

- 思路：当$L(M_1) = \emptyset$时，E_{TM}是EQ_{TM}的特例。
- 设TM R判定EQ_{TM}，构造判定E_{TM}的TM S如下：对于输入$\langle M \rangle$，S执行
  1. 在输入$\langle M, M_1 \rangle$上运行R，其中M_1是拒绝所有输入的TM；
  2. 如果R接受，则接受；如果R拒绝，则拒绝。

定义1 (接受（拒绝）计算历史)

设M是一个图灵机，w是一个串，则称格局序列$C_1, C_2, \cdots, C_l$为接受（拒绝）计算历史，其中C_1是M在w上的起始格局，C_l是接受(拒绝)格局。

定义1 (接受（拒绝）计算历史)

设M是一个图灵机，w是一个串，则称格局序列$C_1, C_2, \cdots, C_l$为接受（拒绝）计算历史，其中C_1是M在w上的起始格局，C_l是接受(拒绝)格局。

- 如果M在w上不停机，则M在w上没有接受计算历史和拒绝计算历史；

定义1 (接受(拒绝)计算历史)

设M是一个图灵机，w是一个串，则称格局序列$C_1, C_2, \cdots, C_l$为接受(拒绝)计算历史，其中C_1是M在w上的起始格局，C_l是接受(拒绝)格局。

- 如果M在w上不停机，则M在w上没有接受计算历史和拒绝计算历史；
- 确定型TM在任何给定输入上至多只有一个计算历史；

定义1（接受（拒绝）计算历史）

设M是一个图灵机，w是一个串，则称格局序列$C_1, C_2, \cdots, C_l$为接受（拒绝）计算历史，其中C_1是M在w上的起始格局，C_l是接受(拒绝)格局。

- 如果M在w上不停机，则M在w上没有接受计算历史和拒绝计算历史；
- 确定型TM在任何给定输入上至多只有一个计算历史；
- 非确定型TM可能有多个计算历史。

定义2 (线性界限自动机(LBA))

不允许读写头离开包含输入的带区域的受限图灵机，

定义2 (线性界限自动机(LBA))

不允许读写头离开包含输入的带区域的受限图灵机，即当读写头试图移出输入的两个端点时，读写头保持原地不动。

- 设LBA M有q个状态、g个带符号，则对于长度为n的带子，M有　　个不同的格局。

定义2 (线性界限自动机(LBA))

不允许读写头离开包含输入的带区域的受限图灵机，即当读写头试图移出输入的两个端点时，读写头保持原地不动。

- 设LBA M有q个状态、g个带符号，则对于长度为n的带子，M有qng^n个不同的格局。

定理5 ($A_{LBA} = \{\langle M, w\rangle | LBA\ M$接受$w\}$是可判定的)

证明.

$\square$

定理5 ($A_{LBA} = \{\langle M, w\rangle | LBA\ M$接受$w\}$是可判定的)

证明.

构造判定A_{LBA}的TM如下：对于输入的$\langle M, w\rangle$，执行

$\square$

定理5 ($A_{LBA} = \{\langle M, w\rangle | LBA\ M$接受$w\}$是可判定的)

证明.

构造判定A_{LBA}的TM如下：对于输入的$\langle M, w\rangle$，执行

1. 在w上模拟M qng^n步，或者直至其停机；

定理5 ($A_{LBA} = \{\langle M, w\rangle | LBA\ M$接受$w\}$是可判定的)

证明.

构造判定A_{LBA}的TM如下：对于输入的$\langle M, w\rangle$，执行

1. 在w上模拟$M\ qng^n$步，或者直至其停机；
2. 如果M停机，则当其接受时接受，其拒绝时拒绝；

□

定理5 ($A_{LBA} = \{\langle M, w\rangle | LBA\ M$接受$w\}$是可判定的)

证明.

构造判定A_{LBA}的TM如下：对于输入的$\langle M, w\rangle$，执行

1. 在w上模拟M qng^n步，或者直至其停机；
2. 如果M停机，则当其接受时接受，其拒绝时拒绝；
3. 如果M不停机，则拒绝。

□

定理6 ($E_{LBA} = \{\langle M\rangle | LBA\ M$满足$L(M) = \emptyset\}$不可判定)

证明.

$\square$

定理6 ($E_{LBA} = \{\langle M\rangle | LBA\ M$满足$L(M) = \emptyset\}$不可判定)

证明.

- 思路：将A_{TM}约化为E_{LBA}

$\square$

定理6 ($E_{LBA} = \{\langle M\rangle | LBA\ M$满足$L(M) = \emptyset\}$不可判定)

证明.

- 思路：将A_{TM}约化为E_{LBA}
  - 对于给定的TM M以及输入串w，构造LBA B，使得可以通过检查$L(B)$是否为空集来判定M是否接受w；

定理6 ($E_{LBA} = \{\langle M\rangle | LBA\ M满足L(M) = \emptyset\}$不可判定)

证明.

- 思路：将A_{TM}约化为E_{LBA}
  - 对于给定的TM M以及输入串w，构造LBA B，使得可以通过检查$L(B)$是否为空集来判定M是否接受w；
  - M接受w

□

定理6 ($E_{LBA}=\{\langle M\rangle | LBA\ M$满足$L(M)=\emptyset\}$不可判定)

证明.

- 思路：将A_{TM}约化为E_{LBA}
  - 对于给定的TM M以及输入串w，构造LBA B，使得可以通过检查$L(B)$是否为空集来判定M是否接受w；
  - M接受$w \Longleftrightarrow$存在接受计算历史s

$\square$

定理6 ($E_{LBA} = \{\langle M\rangle | LBA\ M$满足$L(M) = \emptyset\}$不可判定)

证明.

- 思路：将A_{TM}约化为E_{LBA}
  - 对于给定的TM M以及输入串w，构造LBA B，使得可以通过检查$L(B)$是否为空集来判定M是否接受w；
  - M接受$w \Longleftrightarrow$存在接受计算历史s
  - 构造LBA B，使得B接受接受计算历史s

□

LBA B的构造:

LBA B的构造：对于输入的x，如果x是M在w上的接受计算历史，则B应当接受x。

LBA B的构造：对于输入的x，如果x是M在w上的接受计算历史，则B应当接受x。

设$x = \#C_1\#C_2\#\cdots\#C_l$,$B$检查其是否是接受计算历史：

LBA B的构造：对于输入的x，如果x是M在w上的接受计算历史，则B应当接受x。

设$x = \#C_1\#C_2\# \cdots \#C_l$,$B$检查其是否是接受计算历史：

1. C_1是M在w上的起始格局$q_{ini}w_1 \cdots w_n$

LBA B的构造：对于输入的x，如果x是M在w上的接受计算历史，则B应当接受x。

设$x = \#C_1\#C_2\#\cdots\#C_l$,$B$检查其是否是接受计算历史：

1. C_1是M在w上的起始格局$q_{ini}w_1\cdots w_n$
2. C_{i+1}是C_i的合法结果

LBA B的构造：对于输入的x，如果x是M在w上的接受计算历史，则B应当接受x。

设$x = \#C_1\#C_2\#\cdots\#C_l$,$B$检查其是否是接受计算历史：

1. C_1是M在w上的起始格局$q_{ini}w_1\cdots w_n$
2. C_{i+1}是C_i的合法结果
3. C_l是M的一个接受格局

LBA B的构造：对于输入的x，如果x是M在w上的接受计算历史，则B应当接受x。

设$x = \#C_1\#C_2\#\cdots\#C_l$,$B$检查其是否是接受计算历史：

1. C_1是M在w上的起始格局$q_{ini}w_1\cdots w_n$
2. C_{i+1}是C_i的合法结果
   - 检查C_i中除读写头下位置及其相邻位置外，判定C_i, C_{i+1}是否相同；
3. C_l是M的一个接受格局

LBA B的构造：对于输入的x，如果x是M在w上的接受计算历史，则B应当接受x。

设$x = \#C_1\#C_2\#\cdots\#C_l$,$B$检查其是否是接受计算历史：

1. C_1是M在w上的起始格局$q_{ini}w_1\cdots w_n$
2. C_{i+1}是C_i的合法结果
   - 检查C_i中除读写头下位置及其相邻位置外，判定C_i, C_{i+1}是否相同；
   - 以上特殊位置之间是否满足转移函数要求；
3. C_l是M的一个接受格局

A_{TM}到E_{LBA}的约化.

$\square$

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

$\square$

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

1. 对于输入$\langle M, w\rangle$，构造LBA B；

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

1. 对于输入$\langle M, w\rangle$，构造LBA B；
2. 在输入$\langle B\rangle$上运行R；

$\square$

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

1. 对于输入$\langle M, w\rangle$，构造LBA B；
2. 在输入$\langle B\rangle$上运行R；
3. 如果R拒绝，则接受；如果R接受则拒绝。

$\square$

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

1. 对于输入$\langle M, w\rangle$，构造LBA B；
2. 在输入$\langle B\rangle$上运行R;
3. 如果R拒绝，则接受；如果R接受则拒绝。

- 如果R接受$\langle B\rangle$

□

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

1. 对于输入$\langle M, w\rangle$，构造LBA B；
2. 在输入$\langle B\rangle$上运行R；
3. 如果R拒绝，则接受；如果R接受则拒绝。

- 如果R接受$\langle B\rangle \Longrightarrow L(B) = \emptyset$

□

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

1. 对于输入$\langle M, w\rangle$，构造LBA B；
2. 在输入$\langle B\rangle$上运行R；
3. 如果R拒绝，则接受；如果R接受则拒绝。

- 如果R接受$\langle B\rangle \Longrightarrow L(B) = \emptyset \Longrightarrow M$在$w$上没有接受计算历史

□

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

1. 对于输入$\langle M, w\rangle$，构造LBA B；
2. 在输入$\langle B\rangle$上运行R；
3. 如果R拒绝，则接受；如果R接受则拒绝。

- 如果R接受$\langle B\rangle \Longrightarrow L(B) = \emptyset \Longrightarrow M$在$w$上没有接受计算历史$\Longrightarrow M$不接受$w$

$\square$

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

1. 对于输入$\langle M, w\rangle$，构造LBA B；
2. 在输入$\langle B\rangle$上运行R；
3. 如果R拒绝，则接受；如果R接受则拒绝。

- 如果R接受$\langle B\rangle \Longrightarrow L(B) = \emptyset \Longrightarrow M$在$w$上没有接受计算历史$\Longrightarrow M$不接受$w \Longrightarrow S$拒绝$\langle M, w\rangle$

$\square$

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

1. 对于输入$\langle M, w\rangle$，构造LBA B；
2. 在输入$\langle B\rangle$上运行R；
3. 如果R拒绝，则接受；如果R接受则拒绝。

- 如果R接受$\langle B\rangle \Longrightarrow L(B) = \emptyset \Longrightarrow M$在$w$上没有接受计算历史$\Longrightarrow M$不接受$w \Longrightarrow S$拒绝$\langle M, w\rangle$
- 如果R拒绝$\langle B\rangle$

□

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

1. 对于输入$\langle M, w\rangle$，构造LBA B；
2. 在输入$\langle B\rangle$上运行R；
3. 如果R拒绝，则接受；如果R接受则拒绝。

- 如果R接受$\langle B\rangle \Longrightarrow L(B) = \emptyset \Longrightarrow M$在$w$上没有接受计算历史$\Longrightarrow M$不接受$w \Longrightarrow S$拒绝$\langle M, w\rangle$
- 如果R拒绝$\langle B\rangle \Longrightarrow L(B) \neq \emptyset$

□

A_{TM}到E_{LBA}的约化.

设TM R判定E_{LBA}，构造判定A_{TM}的TM S如下：

1. 对于输入$\langle M, w\rangle$，构造LBA B；
2. 在输入$\langle B\rangle$上运行R；
3. 如果R拒绝，则接受；如果R接受则拒绝。

- 如果R接受$\langle B\rangle \Longrightarrow L(B) = \emptyset \Longrightarrow M$在$w$上没有接受计算历史$\Longrightarrow M$不接受$w \Longrightarrow S$拒绝$\langle M, w\rangle$
- 如果R拒绝$\langle B\rangle \Longrightarrow L(B) \neq \emptyset \Longrightarrow M$接受$w$

□

定义3（可计算函数）

若对于函数$f:\sum^* \to \sum^*$，存在$TM\ M$，使得对每个输入w，M都停机且只有$f(w)$出现在带上，则称f是可计算函数

定义4 (映射可归约)

对于语言A, B，如果存在可计算函数$f : \sum^* \to \sum^*$，使得对于每个w，有

$$w \in A \Longleftrightarrow f(w) \in B$$

则称A映射可归约为B。记为$A \leq_m B$。函数f称为A到B的归约。

例1 (A_{TM}与$HALT_{TM}$间的约化)

例1 (A_{TM}与$HALT_{TM}$间的约化)

- 目标：构造可计算函数f，使得

例1 (A_{TM}与$HALT_{TM}$间的约化)

- 目标：构造可计算函数f，使得

$$\langle M, w\rangle \in A_{TM} \Longleftrightarrow \langle f(M), w'\rangle \in HALT_{TM}$$

例1 (A_{TM}与$HALT_{TM}$间的约化)

- 目标：构造可计算函数f，使得

$$\langle M, w\rangle \in A_{TM} \iff \langle f(M), w'\rangle \in HALT_{TM}$$

- $f(M) = M'$:

例1 (A_{TM}与$HALT_{TM}$间的约化)

- 目标：构造可计算函数f，使得

$$\langle M, w\rangle \in A_{TM} \Longleftrightarrow \langle f(M), w'\rangle \in HALT_{TM}$$

- $f(M) = M'$:
  - 对于输入的x，在x上运行M

例1 (A_{TM}与$HALT_{TM}$间的约化)

- 目标：构造可计算函数f，使得

$$\langle M, w\rangle \in A_{TM} \Longleftrightarrow \langle f(M), w'\rangle \in HALT_{TM}$$

- $f(M) = M'$:
  - 对于输入的x，在x上运行M
  - 如果M接受，则接受

例1 (A_{TM}与$HALT_{TM}$间的约化)

- 目标：构造可计算函数f，使得

$$\langle M, w\rangle \in A_{TM} \Longleftrightarrow \langle f(M), w'\rangle \in HALT_{TM}$$

- $f(M) = M'$:
  - 对于输入的x，在x上运行M
  - 如果M接受，则接受
  - 如果M拒绝，则进入循环

例1 (A_{TM}与$HALT_{TM}$间的约化)

- 目标：构造可计算函数f，使得

$$\langle M, w\rangle \in A_{TM} \Longleftrightarrow \langle f(M), w'\rangle \in HALT_{TM}$$

- $f(M) = M'$:
  - 对于输入的x，在x上运行M
  - 如果M接受，则接受
  - 如果M拒绝，则进入循环
- $w' = w$

定理7

定理7

- 如果$A \leq_m B$且B可判定/可识别，则A也可判定/可识别。

定理7

- 如果$A \leq_m B$且B可判定/可识别，则A也可判定/可识别。
- 如果$A \leq_m B$且A不是可判定/可识别，则B也不是可判定/可识别。

定理7

- 如果$A \leq_m B$且B可判定/可识别，则A也可判定/可识别。
- 如果$A \leq_m B$且A不是可判定/可识别，则B也不是可判定/可识别。
- $A \leq_m B \Longleftrightarrow \overline{A} \leq_m \overline{B}$

定理7

- 如果$A \leq_m B$且B可判定/可识别，则A也可判定/可识别。
- 如果$A \leq_m B$且A不是可判定/可识别，则B也不是可判定/可识别。
- $A \leq_m B \iff \overline{A} \leq_m \overline{B}$
- A_{TM}是图灵可识别的，而$\overline{A_{TM}}$不是图灵可识别的

定理8 (EQ_{TM}既不图灵可识别，也不是补图灵可识别)

定理8 (EQ_{TM}既不图灵可识别，也不是补图灵可识别)

- EQ_{TM}不是图灵可识别的：

定理8 (EQ_{TM}既不图灵可识别，也不是补图灵可识别)

- EQ_{TM}不是图灵可识别的：
  - $\overline{A_{TM}} \leq_m EQ_{TM}$

定理8 (EQ_{TM}既不图灵可识别，也不是补图灵可识别)

- EQ_{TM}不是图灵可识别的：
  - $\overline{A_{TM}} \leq_m EQ_{TM} \Longleftrightarrow A_{TM} \leq_m \overline{EQ_{TM}}$

定理8 (EQ_{TM}既不图灵可识别，也不是补图灵可识别)

- EQ_{TM}不是图灵可识别的：
  - $\overline{A_{TM}} \leq_m EQ_{TM} \Longleftrightarrow A_{TM} \leq_m \overline{EQ_{TM}}$
  - f：对于输入的$\langle M, w\rangle$执行

定理8 (EQ_{TM}既不图灵可识别，也不是补图灵可识别)

- EQ_{TM}不是图灵可识别的：
  - $\overline{A_{TM}} \leq_m EQ_{TM} \Longleftrightarrow A_{TM} \leq_m \overline{EQ_{TM}}$
  - f：对于输入的$\langle M, w\rangle$执行
    1. 构造两个TM M_1, M_2如下

定理8 (EQ_{TM}既不图灵可识别，也不是补图灵可识别)

- EQ_{TM}不是图灵可识别的：
  - $\overline{A_{TM}} \leq_m EQ_{TM} \Longleftrightarrow A_{TM} \leq_m \overline{EQ_{TM}}$
  - f：对于输入的$\langle M, w\rangle$执行
    1. 构造两个TM M_1, M_2如下

       M_1:对任何输入都拒绝

定理8 (EQ_{TM}既不图灵可识别，也不是补图灵可识别)

- EQ_{TM}不是图灵可识别的：
  - $\overline{A_{TM}} \leq_m EQ_{TM} \Longleftrightarrow A_{TM} \leq_m \overline{EQ_{TM}}$
  - f：对于输入的$\langle M, w\rangle$执行
    1. 构造两个TM M_1, M_2如下

       M_1:对任何输入都拒绝

       M_2:对任何输入x，在w上运行M，如果M接受，则M_2接受x

定理8 (EQ_{TM}既不图灵可识别，也不是补图灵可识别)

- EQ_{TM}不是图灵可识别的：
  - $\overline{A_{TM}} \leq_m EQ_{TM} \Longleftrightarrow A_{TM} \leq_m \overline{EQ_{TM}}$
  - f：对于输入的$\langle M, w\rangle$执行
    1. 构造两个TM M_1, M_2如下
       M_1:对任何输入都拒绝
       M_2:对任何输入x，在w上运行M，如果M接受，则M_2接受x
    2. 输出$\langle M_1, M_2\rangle$

- $\overline{A_{TM}} \leq_m \overline{EQ_{TM}}$

- $\overline{A_{TM}} \leq_m \overline{EQ_{TM}} \Longleftrightarrow A_{TM} \leq_m EQ_{TM}$

- $\overline{A_{TM}} \leq_m \overline{EQ_{TM}} \Longleftrightarrow A_{TM} \leq_m EQ_{TM}$
- g:对于输入的$\langle M, w \rangle$执行

- $\overline{A_{TM}} \leq_m \overline{EQ_{TM}} \Longleftrightarrow A_{TM} \leq_m EQ_{TM}$
- g:对于输入的$\langle M, w\rangle$执行
  - 构造两个TM M_1, M_2如下:

- $\overline{A_{TM}} \leq_m \overline{EQ_{TM}} \Longleftrightarrow A_{TM} \leq_m EQ_{TM}$
- g:对于输入的$\langle M, w\rangle$执行
  - 构造两个TM M_1, M_2如下：
    - M_1:对任何输入都接受;

- $\overline{A_{TM}} \leq_m \overline{EQ_{TM}} \Longleftrightarrow A_{TM} \leq_m EQ_{TM}$
- g:对于输入的$\langle M, w\rangle$执行
  - 构造两个TM M_1, M_2如下：
    - M_1:对任何输入都接受；
    - M_2:对任何输入x，在w上运行M，如果M接受，则M_2接受x

- $\overline{A_{TM}} \leq_m \overline{EQ_{TM}} \Longleftrightarrow A_{TM} \leq_m EQ_{TM}$
- g:对于输入的$\langle M, w\rangle$执行
    - 构造两个TM M_1, M_2如下：
        - M_1:对任何输入都接受；
        - M_2:对任何输入x，在w上运行M，如果M接受，则M_2接受x
    - 输出$\langle M_1, M_2\rangle$