

形式语言与自动机 期末复习

XY

6.12

Lecture 6 正规语言的性质与运算

Thm (Pumping 引理) ← 必要条件！判定“不是”

设 L 为正规语言，存在常数 $n \geq 1$ ，使得任一长度 $\geq n$ 的字符串 $w \in L$, $|w| \geq n$, 都可以分成三个部分，即 $w = xyz$. 且满足 1. $y \neq \epsilon$ 2. $|xy| \leq n$ 3. 对任何 $k \geq 0$, 有 $xy^k z \in L$.

记：取 $n=10$. 设 L 为 DFA $D=(Q, \Sigma, S, q_0, F)$ 的语言，用鸽巢

证明不正规： 考虑 $\forall n, \exists w, s.t |w| \geq n$, 且对 $\forall x, y, z$ 满足 $w = xyz$, $y \neq \epsilon$, $|xy| \leq n$, 能够找到一个 k , 使 $xy^k z \notin L$.

e.g. $\{0^m 1^m\}$. $w = 0^n 1^n = xyz \Rightarrow xy$ 都是 0串 $\Rightarrow xz$ 中 1 有 n 个, 0 < n 个 $\notin L \Rightarrow L$ 非正规

Thm (判定正规语言是否为空) $L = \emptyset$.

① 以 DFA 表示：初态出发是否可到达某一状态 $O(n^2)$ $\{R_1 + R_2\}$ 是 iff $\{R_1\}$ 和 $\{R_2\}$ 是

② 以正规表达式表示：归纳基： $L(\emptyset)$ 是, $L(\epsilon)$, $L(A)$ 不是 $O(n^2)$ $\{L(R^*)\}$ 一定是、括号不变是否

Thm (判定是否包含特定字符串)

转 DFA 直接模拟，或转 ϵ -NFA / NFA 模拟 $O(n^2)$ $\{DFA\}$ $\{NFA\}$ $O(n^2)$, S 为状态数

Thm (判定语言是否相等)

转两个无重名状态的 DFA，合并成新 DFA，用图表示，若两初态不可区分则相等。即语言复杂度 $O(n^4) \rightarrow O(n^2)$

正规语言的封闭运算

① 并. 若 L, M 正规语言，则有正表 R, S 使 $L(R) = L, L(S) = M$.

由正表定义，有 $L(R+S) = L(R) \cup L(S) = L \cup M$. 因此 $L \cup M$ 为正规语言.

② 星闭包. $L(R) = L \Rightarrow L(R^*) = (L(R))^* = L^*$.

③ 连乘. $L(R) = L, L(S) = M \Rightarrow L(RS) = L(R)L(S) = LM$.

④ 补. 若 L 是 Σ 上正语，则 $\overline{L} = \Sigma^* - L$ 也是正语。用 DFA，终态和非终态互换。

⑤ 交. $L \cap M = \overline{\overline{L} \cup \overline{M}}$. 也可构造 $A = (Q \times Q_M, \Sigma, S, \langle q_L, q_M \rangle, F_L \times F_M)$. P29-30. 例.

⑥ 差. $L - M = L \cap \overline{M}$

⑦ 反向. $L^R = \{w^R \mid w \in L\}$ |
设 L 对应正表为 E . 构造 E^R . 归纳设 $L(E^R) = L^R$. 则 L^R 正规

|
法一. 构造 DFA，原 L 的逆反向基础： $\emptyset, \Sigma, \alpha$ 成立. 归纳 $\{E = E_1 + E_2, E^R = E_1^R + E_2^R, L(E^R) = L^R\}$

初态作为唯一终态，新加 p_0 为初态，将每个终态加 $E_1, E_2, E_1^R, E_2^R, \dots$

Σ 转移弧，构造新 ϵ -NFA 使 $L^R = L^R$ P34 例 $E_1^*, (E_1^R)^*, (E_1), (E_2^R), \dots$

⑧ 同态. 设映射 $h: \Sigma \rightarrow T^*$, 则对 $w = a_1 a_2 \dots a_n \in \Sigma^*$. 定义 $h(w) = h(a_1) \dots h(a_n)$. 归纳造 $h(E)$.

$h(w) = h(a_1) \dots h(a_n)$. 称为串 w 的同态. \rightarrow (归纳证 $L(h(E)) = h(L(E)) = h(L)$)

对语言 $L \subseteq \Sigma^*$. L 的同态 $h(L) = \{h(w) \mid w \in L\}$ 也是正规语言。

⑨ 反同态. $h: \Sigma \rightarrow T^*$. 对语言 $S \subseteq T^*$. 定义 $h'(S) = \{w \mid w \in \Sigma^* \wedge h(w) \in S\}$ 可以不

构造 DFA. B. 例如 w , B 每条转移边对应 $S \subseteq T^*$ 的 DFA A 里 $h(a)$ 的转移。是一对一

即 $A = (Q, T, S, q_0, F)$. $B = (Q, \Sigma, Y, q_0, F)$. $Y(q, a) = S'(q, h(a)) \rightarrow$ 一次性走完

例子：P40. 复习. P41. 典型题 $h(a)$ 整个字符串

△用封闭运算证明不是正规语言.

e.g. $L = \{a^i b^j c^k \mid i, j, k \geq 0, i=1 \text{ 时 } j=k\}$, 既然 ab^*c^*

反证. 若 L 正规. 又: $\{ab^j c^k \mid j, k \geq 0\}$ 正规. 那么 $L \cap \{\cdots\} = \{ab^n c^n \mid n \geq 0\} = L'$

依 $h(a) = \Sigma, h(b) = 0, h(c) = 1$, 则 $h(L') = \{0^n 1^n \mid n \geq 0\}$ 正规. 但不是. 方框 (Pumping)

Lecture 7 下推自动机.

Def (下推自动机)

(带有堆栈的 DFA) \Rightarrow PDA 七元组 $P = (Q, \Sigma, I, \delta, q_0, Z_0, F)$

当前格局用 (q, w, γ) 表示. 称为 ID.
状态 留余输入串 栈中内容 (底顶在左)

$(q, aw, X\beta) \vdash (p, w, \alpha\beta) \text{ iff } (p, \alpha) \in \delta(q, a, X)$. \Rightarrow 自底递归闭包
其中 $p, q \in Q, a \in \Sigma \setminus \{\epsilon\}, w \in \Sigma^*, X \in I, \alpha, \beta \in I^*$.

Def (下推自动机的语言)

① 终态接受 $L(P) = \{w \mid (q_0, w, Z_0) \xrightarrow{*} (q, \epsilon, \alpha) \text{ 其中 } q \in F, \alpha \in I^*\}$.

② 空栈接受 $N(P) = \{w \mid (q_0, w, Z_0) \xrightarrow{*} (q, \epsilon, \epsilon) \text{ 其中 } q \in Q\}$

Thm (两种定义的等价性)

① 空栈 \rightarrow 终态

$P_N = (Q, \Sigma, I, \delta_N, q_0, Z_0) \Rightarrow P_F = (Q \cup \{p_0, p_f\}, \Sigma, I \cup \{X_0\}, \delta_F, p_0, X_0, \{q_f\})$

1. $\delta_F(p_0, \epsilon, X_0) = \{(q_0, Z_0 X_0)\}$. 开始加 X_0 .

2. $\forall q \in Q, \forall a \in I, \forall Y \in \Sigma: \delta_F(q, a, Y) = \delta_N(q, a, Y)$

3. $\forall q \in Q: \delta_F(q, \epsilon, X_0) = \{(p_f, \epsilon)\}$. 消到空栈只剩 X_0 . 可转移到 q 状态.

② 终态 \rightarrow 空栈

$P_F = (Q, \Sigma, I, \delta_F, q_0, Z_0, F) \Rightarrow P_N = (Q \cup \{p_0, p\}, \Sigma, I \cup \{X_0\}, \delta_N, p_0, X_0)$

1. $\delta_N(p_0, \epsilon, X_0) = \{(q_0, Z_0 X_0)\}$. 开始加 X_0 .

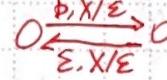
2. $\forall q \in Q, a \in I, Y \in \Sigma: \delta_N(q, a, Y) = \delta_F(q, a, Y)$

3. $\forall q \in F, \forall Y \in I \cup \{X_0\}: \delta_N(q, \epsilon, Y) = \{(p, \epsilon)\}$. P_F 接受则可转移到 p .

4. $\forall Y \in I \cup \{X_0\}: \delta_N(p, \epsilon, Y) = \{(p, \epsilon)\}$. P 不断弹出栈顶到空.

构造 PDA. P23. 弹出两个 X 的处理: 多加一节点.

P24. $\{ww\} \cap ab^*$, a 与 b 数量不同 | P25.



Lecture 8 上下文无关文法 \Leftrightarrow 下推自动机.

Thm. (上下文无关文法与下推自动机的等价性) 站字表 工成能

① CFG $G = (V, T, P, S)$ 构造 PDA $E_N = (f(q), T, V \cup T, \delta, q, S)$

非终终带标注 (1) 对每一个 $A \in V$. $\delta(q, \epsilon, A) = f(q, \beta) \mid A \rightarrow \beta \in P$. 非终: 只推

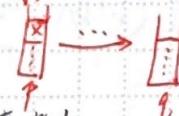
(2) 对每一个 $a \in T$. $\delta(q, a, a) = f(q, \epsilon)$ 终结符: 已遍历消掉且标记

6.11

证明: P8~P9.

→ 定义接受

构造符

② PDA $E = (\mathcal{Q}, \Sigma, \delta, S, q_0, Z_0)$ 构造 CFG $G = (V, \Sigma, P, S)$.产生集合 P 定义如下:(1) 对每一个 $p \in \mathcal{Q}$, G 包含产生式 $S \rightarrow [q_0 Z_0 p]$: 定义接受 q_0 出发消完 Z_0 后到任意节点 p .(2) 若 $(q, X_1 X_2 \dots X_k) \in \delta(p, a, X)$, 则有产生式 $[p X p_k] \rightarrow a [q X_1 p_1] [p_1 X_2 p_2] \dots [p_{k-1} X_k p_k]$. 其中, p_i 任意节点.注意不是 p_j . $= \{S\} \cup \{[p X q] \mid p, q \in \mathcal{Q} \wedge X \in I\}$ 含义: P 消完栈顶 X 后到达 q . $a \in \Sigma \text{ 或 } a = \epsilon$.得: $(q, \epsilon) \in \delta(p, a, X) \Rightarrow [p X q] \rightarrow a$: S 右边是 (q, ϵ) 因此没有中间点可跳.左边必须到状态 q . 若 $a = \epsilon$, 那么 $[q X q] \rightarrow \epsilon$.对 $q, p \in \mathcal{Q}, X \in I$. $(q, w, X) \vdash (p, \epsilon, \epsilon)$ iff $[q X p] \Rightarrow w$.

P11 P13.

证明: P19-21.

6.15

Lecture 9 确定下推自动机.

Def (确定下推自动机) (DPDA)

(1) 对于 $a \in \Sigma$ 或 $a = \epsilon$, $X \in I$. $\delta(q, a, X)$ 最多包含一个元素(2) 对于 $a \in \Sigma$. 若 $\delta(q, a, X) \neq \emptyset$ (空集). 则 $\delta(q, \epsilon, X) = \emptyset$.e.g. P5-8. $L = \{wwr \mid w \in \{0,1\}\}$ 有一个非确定 PDA 是不可能 DPDA? $L = \{wcw^R \mid \dots\}$ 可构造 DPPA (中间转移确定)例. P11. 注意 DPDA 对于不同情况 ($n=0$ 等) 的分类讨论办法.

△ 确定下推自动机 ⊂ 正规语言 DPDA 计算能力大于正规语言!

若 L 为正规语言, 则存在 DPDA P , 使 $L(P) = L$.Prof. 构造 DFA $A = (\mathcal{Q}, \Sigma, \delta_A, q_0, F) \Rightarrow$ DPDA $P = (\mathcal{Q}, \Sigma, \{Z_0\}, \delta_P, q_0, Z_0, F)$ (忽略 PDA 的栈) (可规约证相等) $\delta_P(q, a, Z_0) = f(p, Z_0)$ iff $\delta_A(q, a) = p$.反之不成立. e.g. $\{wcw^R\}$ 不是正规语言 (可 Pumping 证)

Def (前缀性质) 注意“前缀性质”反而是不存在两两间前缀关系.

语言 L 具有前缀性质 iff 不存在 $x, y \in L$, $x \neq y$, 且 x 为 y 的前缀Thm. 一个语言 L 是某个 定义接受 的 DPDA P 的语言 iff L 具有前缀性质, 且是某个 DPDA P 的语言.

△ 确定下推自动机 ⊂ 上下文无关语言.

某些 CFL 不是任何 DPDA 的语言. e.g. $L_{wur} = \{wwr\}$ 不是任何 DPDA 的语言.Def (确定的上下文无关语言). 若 L 是某个 DPDA 的语言, 则 L 是 ~

△ 确定下推自动机 ⇒ 无二义文法.

可构造 DPDA → CFG, 也有唯一最左推导.

若 L 可被 定义接受 的 DPDA P 表示, 则 $L = N(P)$, 则 L 存在一个无二义文法.

P24 批量 + \$ → 有向图 → 文法 → CFG 去掉 \$

→ 固有二义的语言不是任何DPDA的语言. e.g. $L_1 = \{a^n b^n c^n d^n \mid n, m \geq 1\} \cup \{a^n b^m c^n d^m \mid n, m \geq 1\}$.
 $L_2 = \{a^i b^j c^k \mid i, j, k \geq 0, i=j \text{ 或 } j=k\}$.

← 存在非固有二义的语言, 不是任何DPDA的语言. e.g. $\{wwr^k\}$.

* $CFG = \text{空栈PDA} < \text{终态PDA} > \text{无二义的CFG} > \text{终态DPDA} = \text{确定的CFG}$

中间: (DPDA空栈) \leftrightarrow DFA, RE, NFA, ϵ -NFA (外层: 终态DPDA) $\xrightarrow{\text{为什么终态DPDA?}} >$ 空栈DPDA?

Lecture 10 CFG 的简化及 Chomsky 范式.

Def (有用符号、无用符号).

即参与过完整推导.

$x \in VUT$ 是有用的 iff $S \xrightarrow{*} \alpha x \beta \xrightarrow{*} w$, 其中 $w \in T^*$, $\alpha, \beta \in (VUT)^*$

Def (生成符号、可达符号) 注意“生成”指可以生成串, 不是可以从 S 生成 (是可达!).

$\exists w \in T^*, \text{ s.t. } X \xrightarrow{*} w \quad \exists \alpha, \beta \in (VUT)^*, \text{ s.t. } S \xrightarrow{*} \alpha X \beta$. 有用符号必为生成. 可达符号
反之未 $S \xrightarrow{*} AB \mid A$

Thm ① 消去非生成 ② 消去不可达

剩余皆为有用符号, 且文法与原来等价.

B 生成 & 可达, 但无用.

△ 计算生成/可达符号集: 归纳递推, trivial 注意生成 可达! $S \xrightarrow{*} AB \Rightarrow Ab$.

Def (可致空符号)

需产生式右部都是生成左才是

$A \in V$ 是可致空的 iff $A \xrightarrow{*} \epsilon$.

△ 计算可致空符号集. 同上归纳递推. 右每个都可致空 \rightarrow 左可致空

Thm. 消去 ϵ 产生式.

$\cdot A \rightarrow A_1 A_2 \dots A_k$ 对应一组新产生式, 其中每个可致空符号可出现/不出现, 共 2^m 个产生式 (共有 m 个).

\cdot 去掉所有 $A \rightarrow \epsilon$ 产生式.

若 $m=k$ 则是 2^m-1 个.

\Rightarrow 得到的新语言为 $L - \{\epsilon\}$. ~~为什么没有 ϵ 产生式?~~

Def (Unit 产生式) 形如 $A \rightarrow B$ 的产生式. $A, B \in V$.

Def (Unit 偶对) $A, B \in V$, 称 (A, B) 为 ~ iff $A \xrightarrow{*} B$, 且推导过程仅使用 Unit 产生式.

△ 计算 Unit 偶对集 基础: (A, A) 是 归纳: 若 (A, B) 是, $B \rightarrow C$ 是 Unit 产生式, 则 (A, C) 是.

Thm. 消去 Unit 产生式.

对每个 Unit 偶对 (A, B) , 在 G_i 中加入 $A \rightarrow a$, 其中 $B \rightarrow a$ 为非 Unit 产生式. $\Rightarrow L(G_i) = L(G)$

(1) 先消 ϵ 产生式 (2) 消 Unit 产生式 (3) 消无用符号 (非生成 + 不可达).

Def (乔姆斯基范式) Chomsky Normal Form \rightarrow CNF

任何不含 ϵ 的非空 CFL 都存在一个 CFG G , 其产生式具有如下两形式之一:

1. $A \rightarrow BC$. A, B, C 均为非终结符

2. $A \rightarrow a$. A 为非终结符, a 是终结符

且 G 中不含无用符号. 这样的文法形式称为 Chomsky 范式.

(4) 若终结符 a 出现在右边长度 > 1 的产生式中, 则引入非终 A 替换, 增加 $A \rightarrow a$.

(5) 将右边长度 > 2 的产生式用归联方法转化为只包含两个非终

△ CKY 算法：给定一个 CFG 和一个串，判断串是否属于 CFG.

每次填一格代表 $[i:j]$ 字符串，枚举中间点 k 后再枚举 $[i:k]$ 与 $[k:j]$ 中填的非终看是否有非终生成它们两个，填入 $[i:j]$.

Lecture 11 上下文无关语言的性质与运算.

6.17

Thm 上下文无关语言的 Pumping 特性

Prof. P5-P7 证明: $S \rightarrow uAy$, $A \rightarrow vAx$, $A \rightarrow w$ $\Rightarrow uv^iwx^jy$.

设 L 是 CFL，则存在正整数 n , s.t. $\forall z \in L$, $|z| \geq n$ 这样串都可分成三部分,

$z = uvNxy$, 满足: ① $v \neq \epsilon$ ② $|vwx| \leq n$ ③ 对 $\forall k \geq 0$, 有 $uv^kwx^ky \in L$.

不存在 $Uniz$ 式. 鸽先

证明不是 CFL: 考虑一个 n , 找一个 $\exists z \in L$, $|z| > n$. 任选 $\forall z = uvNxy$ ($v \neq \epsilon$, $|vwx| \leq n$).

e.g. P13-14. 找 k , 使 $\dots \notin L$.

Thm 上下文无关语言不可判定的性质:

文法无二义. 因有二义语言. 两语言相交是否为空. 两语言是否相等. 语言是否等于 Σ^*

正规)

△ 上下文无关语言的封闭运算

(1) 替换 映射 $s: \Sigma \rightarrow L$ 称为 Σ 上的一个替换, L 为语言集合.

- 对 $a \in \Sigma$, $s(a) = La \in L$. 某个语言.
- 对 $w = a_1a_2 \dots a_n \in \Sigma^*$, $s(w) = s(a_1)s(a_2) \dots s(a_n)$ 语言的拼接 \Rightarrow 语言.
- 对 Σ 上的语言 L , $s(L) = \bigcup_{w \in L} s(w)$. 语言的(拼接的)并 \Rightarrow 语言.

\Rightarrow 若 L 为 Σ 上 CFL, s 为 Σ 上一替换, 且 $\forall a \in \Sigma$, $s(a)$ 也为 CFL, 则 $s(L)$ 也是 CFL.

(2) 并. L, M 为 CFL, $L \cup M$ 为 CFL.

Prof. 替换 s : $s(\emptyset) = L$, $s(1) = M$. 则 $s(\{0,1\}) = L \cup M$. 且 $\{0,1\}$ 为 CFL $\Rightarrow L \cup M \dots$

(3) 闭包 (星~和正~). L 为 CFL, 则 L^* 和 L^+ 也是 CFL.

Prof 替换 s : $s(1) = L$, 则 $s(\{1\}^*) = L^*$, $s(\{1\}^+) = L^+$. 而 $\{1\}^* \cdot \{1\}^+ \subseteq L$ 为 CFL $\Rightarrow \dots$

(4) 连接. L, M 为 CFL, LM 为 CFL.

Prof 替换 s : $s(\emptyset) = L$, $s(1) = M$. 则 $s(\{0,1\}) = LM$. 而 $\{0,1\}$ 为 CFL $\Rightarrow LM \dots$

(5) 同态. $h: \Sigma \rightarrow T^*$, 则对 $w = a_1a_2 \dots a_n \in \Sigma^*$, 定义 $h(w) = h(a_1)h(a_2) \dots h(a_n)$. $\Rightarrow w$ 的同态
对语言 $L \subseteq \Sigma^*$. 定义 L 的同态 $h(L) = \{h(w) | w \in L\}$

Prof. $s(a) = \{h(a)\}$. 则 $s(L) = h(L)$.

(6) 反向. $L^R = \{w^R | w \in L\}$ 也为 CFL. Prof. 构造 CFL' $P' = \{A \rightarrow a^R | A \rightarrow a \in P\}$.

不封闭的: (7) 反同态. $h'(L) = \{w | w \in \Sigma^* \wedge h(w) \in L\}$. P33. ???

(1) 交. $L = \{0^n1^n2^n\}, M = \{0^i1^i2^n\} \Rightarrow L \cap M = \{0^n1^n2^n\}$

(2) 补. $L \cap M = \overline{L \cup M}$. 故补不封闭.

(3) 差. $L = \Sigma^* - L$. 故差不封闭.

(1) CFL 与正规语言的交 $L \cap R$ 为 CFL.

Def. 构造 PDA (终态接受) $P' = (\{Q_p \times Q_A\}, \Sigma, T, S, (q_p, q_A), Z_0, F_p \times F_A)$

Lecture 12 图灵机与递归可枚举语言.

Def (图灵机) $M = (Q, \Sigma, T, \delta, q_0, B, F)$.

有限带符号集 空白符

$S: Q \times T \rightarrow Q \times T \times \{L, R\}$.

$q_0 \in Q, \Sigma \subseteq T, B \in T - \Sigma, F \subseteq Q$

字母表必须在带符号集中!

Def. (用 ID 表达当前格局) $X_1 X_2 \dots X_{i-1} q X_i X_{i+1} \dots X_n$ 称为 ID.

1. $q \in Q$ 为当前 M 状态 2. 带头正扫描 X_i 3. X_1 与 X_n 为最左(右)的非空符.(但 q 可以在带符号集中)

有空符, 例如 $XXqB$.

△ 图灵机: 每次将当前格符号修改后往左/右移一格, 转移状态点 δ .
 t_M 自反传递闭包 t_M^* (或 t^*).

Def (递归可枚举语言) 可以被图灵机接受的语言.

$$L(M) = \{w \mid w \in \Sigma^* \wedge \exists p (p \in F \wedge \alpha \in T^* \wedge \beta \in T^+ \wedge q_0 w t_m^* \alpha p \beta)\}$$

Def (停机) 图灵机不存在下一个移动

任给 M, 容易构造 M' , 使 $L(M) = L(M')$ 且若 $w \in L(M)$ 则对于 w, M' 接受 w 并一定停机.

以后若无特殊说明, 总假定图灵机到达终态后一定停机.

进阶

Def (递归语言) 为什么这样叫? \Rightarrow 递归: 可判断一个元素是否在集合里. (A 与 A 询问可枚举).

称 L 是 ~ iff 存在图灵机 M, s.t. $L = L(M)$ 满足 递归可枚举: 不知道以后会不会枚举到, 不可判断.

1. 若 $w \in L(M)$, 则对于 w, M 接受 w (自动停机) 就是都工作机.

2. 若 $w \notin L(M)$, 对于 w, M 最终也停机.

\Rightarrow 递归语言对应的问题是可判定的?

图灵机的编程技巧.

1. 状态带有储存区. (不是带头!) 状态为二元组. q, I

2. 多道 带符号, 多元组

3. 子例程

对基本图灵机的扩展

1. 多带图灵机: 多条带, 可单独移动. 倍方接受能力与图灵机等价

可用带储存区, 有 2 条道的图灵机模拟每条带: 一道带头位置 - 一道

2. 非确定图灵机: 下一个动作有多种选择. 与图灵机等价. 可用 BFS 模拟功空间树
 受限的图灵机.

1. 具有半无穷带的 ~. 与图灵机等价. 可用双道的半无穷模拟无穷.

2. 多核机: 具有多个下推栈的 PDA.

3. 计数器机: 具有两个计数器的 PDA. 有非负整数 $0, 1, 2, \dots$ (不能 -1) 将线性符号编码
 一个计数器: DPDA. 多个: 图灵机 3 变 2: $P91, M = 2^1 3^j 5^k \leftarrow$ 两个计数器: 下推栈 - 辅助 $x, +$

普通计算机 \leftrightarrow 图灵机

7

Lecture 13 计算理论初步.

Def (图灵机与输入中的二进制编码)

① 图灵机: 假设输入字母表 $\{0,1\}$, 有限状态 q_1, \dots, q_k , 初态 q_1 , 终态 q_f . 只需一个状态 \rightarrow 图灵停机.

注意下脚注: 假设带符号 $x_1 \dots x_m$, x_1 总代表0, x_2 代表1, x_3 代表B; 移动方向D, D₂代表L, R.
从1开始. △ 转移规则 $S(q_i, x_j) = (q_k, x_L, D_m)$ 编码为 $0^i 1 0^j 1 0^k 1 0^l 1 0^m$

△ 图灵机编码为所有转移规则连在一起, 形如 $G_1 | G_2 | G_3 | \dots | G_m | G_n$

② 01字符串: w用1w编码以区分前缀0. (eg. 0 \rightarrow 10, 00 \rightarrow 100) 因此w可对应整数j.

△ 若图灵机的二进制编码为 w_i , 而 w_i 为第j个0串, 称为第j个图灵机, 称为第j个字符串.

③ 图灵机与输入串偶对的编码: $(M, w) \rightarrow C, C' \rightarrow C | | C'$ 也是从1开始.

Def (对角语言)

对于不对应任何图灵机的整数j, 定义 M_j 为不接受任何串的图灵机, 即 $L(M_j) = \emptyset$.

这样, 可规定对 $\forall i \geq 1$, 第i个图灵机为 M_i , 定义对角语言为: $L_d = \{w_i \mid w_i \notin L(M_i)\}$

Thm: L_d 不是递归可枚举语言

反证. 若存在 M_k , st. $L(M_k) = L_d$. 考虑 w_k 是否属于 L_d ? $w_k \notin L(M_k) \Rightarrow w_k \in L_d = L(M_k)$

$w_k \in L(M_k) \Rightarrow w_k \notin L_d = L(M_k)$

Thm: 递归语言的补运算是封闭的

若L是递归语言, 则 \bar{L} 也是.

Def: 对总会停机的图灵机M, 可构造 \bar{M} , 使 $\bar{L} = L(\bar{M})$.

M 的终态变成不可的非终, 无进一步转移; 新增终态Y. (无转移)

对每个非终 q , 带符X, 若无 $S(q, X)$ 定义, 则增加 $S(q, X) = (f, Y, D)$, Y, D可任取.

(把 M 中非终节点所有可能停机的状态导到Y节点, 然后也停机)

Thm: 递归可枚举语言的补运算是封闭的

通用语言 L_u 递归可枚举, 但 \bar{L}_u 不是

进一步地, 我们有 Thm: (若 L 与 \bar{L} 都递归可枚举, 则 L 与 \bar{L} 都是递归的)

图灵机 $L = L(M_1), \bar{L} = L(M_2)$, 用 M_1 并行执行 M_1 和 M_2 .

无论是否 $w \in L$, 均能停机. $w \xrightarrow{\bar{M}_1} \text{Accept} \xrightarrow{\bar{M}_2} \text{Accept} \rightarrow \text{Accept}$
 $w \xrightarrow{\bar{M}_2} \text{Accept} \xrightarrow{\bar{M}_1} \text{Accept} \rightarrow \text{Reject}$

Def (通用语言)

编码 (M, w) 的所有0串集合为 L_u . 其中 (M, w) 满足 $w \in L(M)$ 多带可构造

即: 对偶对 (M, w) , $w \in L(M)$ 且 通用图灵机 U 接受编码后的 (M, w) (不细说)

Thm: (L_u 递归可枚举但不可判定) $\Rightarrow \bar{L}_u$ 不是递归可枚举的

反证. 否则 \bar{L}_u 也可递归, 那么 L_d 也将是递归语言的结果, 但其不是递归(可枚举)的

$w \xrightarrow{L_u} \text{Copy} \xrightarrow{L_u} w | | w \xrightarrow{\text{"M for } \bar{L}_u"} \text{Accept} \xrightarrow{\text{Accept}} \text{Accept}$
 $w \xrightarrow{L_u} \text{Copy} \xrightarrow{L_u} w | | w \xrightarrow{\text{"M for } \bar{L}_u"} \text{Reject} \xrightarrow{\text{Reject}} \text{Reject}$

考虑w是否是合法图灵机编码的两种情况均成立.

△ 语言 = 问题 任给串 $w \in \Sigma^*$, 判定 $w \in L$ 是否成立?

△ 问题的判定 问题对应的语言: 递归 \Rightarrow 可判定, 否则 \Rightarrow 不可判定

递归可枚举 \Rightarrow 部分可判定, 否则 \Rightarrow 非部分可判定

Def. 固灵机停机问题 $L_H = \{C \mid M(C) \text{ 停机}\}$ 对于输入串 C' , 固灵机 C 将停机

L_H 可归约为此问题 (将 M 改成不接受 W 时都不停机) $\Rightarrow L_H$ 不是递归语言

Def. 问题的归约 若能找到算法将问题 P_1 的实例转化成 P_2 的且回答相同, 则称 ~

若 P_1 可归约到 P_2 , 且 P_2 可判定, 则 P_1 可判定; P_2 部分可判定, 则 P_1 也 ~

若 P_1 不可判定, 则 P_2 也 ~; P_1 非部分 ~ P_2 也 ~

△ 判定固灵机语言是否非空、部分可判定. $L_{ne} = \{M \mid L(M) \neq \emptyset\}$

为空: 部分可判定 $L_e = \{M \mid L(M) = \emptyset\}$.

Thm. (Rice 定理) 有关递归可枚举语言的任何非平凡性质都是不可判定的.

性质 (property): $P \subseteq L$ (所有递归语言集合). 若 $P \neq \emptyset$ 或 L , 则不平凡

\Rightarrow 固灵机可接受的语言 L 是否非空/为空/正规语言/上下文无关语言? X

Thm. Post 对应问题. PCP 不可判定 (可将 L_H 归约到 PCP)

$A = w_1, w_2, \dots, w_k \quad B = x_1, x_2, \dots, x_k \quad \Rightarrow \text{CFG 是否歧义不可判定}$

PCP 的该实例有解 $\Leftrightarrow \exists$ 整数序列 i_1, i_2, \dots, i_m , 使 $w_{i_1}, w_{i_2}, \dots, w_{i_m} = x_1, x_2, \dots, x_m$

P. NP 问题. P20

固灵机的时间复杂度: 最多 $T(n)$ 步停机.

非确定固灵机的 ~

问题类 P : 存在固灵机 M , s.t. $L = L(M)$ 且 $T(n)$ 为多项式.

NP 非确定.

$P \subseteq NP \checkmark \quad P = NP ?$

多项式时间归约 P_1 到 P_2

NP-完全问题: ① P 是 NP 问题且任何 NP 问题 P' 可在多项式时间归约到 P .

NP-难问题: 若可证明 P 满足上述①但无法证②, 则称 P 是 ~.

可满足性问题 SAT: 任给 bool 表达式是否可满足?

Cook 定理: SAT 是 NP-完全问题.