华东师范大学
软硬件协同设计技术与应用教育部工程研究中心
URL—http://faculty.ecnu.edu.cn/chenyixiang
yxchen@sei.ecnu.edu.cn

计算理论–The Theory of Computation

陈仪香, 吴恒洋

MoE Engineering Center for Software/Hardware Co-Design Technology and Application
Software Engineering Institute
East China Normal University(ECNU)
Shanghai, China

2018级研究生软件工程理论课程，2018年5月

Outline

- 语言与自动机
- 图灵机
- 计算复杂性
- 时间自动机

语言与自动机

- 正规语言与有限自动机
- 上下文无关语言与下推自动机

有限状态自动机

定义　形式化定义

有限（状态）自动机(finite automaton) 是一个5元组$(Q, \Sigma, \delta, q_0, F)$,其中:

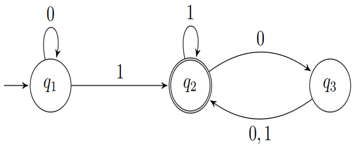
1. Q是一个有限集，其元素称为状态(states),
2. Σ 是一个有限集，其元素称为字符(alphabet),
3. $\delta : Q \times \Sigma \to Q$称为转移函数,
4. $q_0 \in Q$是初始状态,
5. $F \subseteq Q$是接受(终止)状态集.

有限状态自动机

例子：

1. 有限自动机$M_0 = (Q, \Sigma, \delta, q_1, F)$,其中$Q = \{q_1, q_2, q_3\}$, $\Sigma = \{0, 1\}$, $F = \{q_2\}$,δ 定义为：

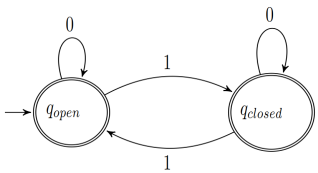
	0	1
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_2	q_3



Example 1

2. 定义自动机$M_{OC} = (\{q_{Open}, q_{Closed}\}, \delta, \{0,1\}, q_{Open}, \{q_{Open}, q_{Closed}\})$,其中$\delta$定义为

	0	1
q_{Open}	q_{Open}	q_{Closed}
q_{Closed}	q_{Closed}	q_{Open}



Example 2

有限状态自动机

定义　输入串/字

自动机$M = (Q, \Sigma, \delta, q_0, F)$的输入串是一个有限字母序列$a_1, a_2, ..., a_n$，其中$\forall i \in I, a_i \in \Sigma$. 符号$\Sigma^*$表示自动机$M$的所有输入串/字集合.

定义　执行

自动机$M = (Q, \Sigma, \delta, q_0, F)$在输入串$w = a_1, a_2, ..., a_n \in \Sigma^*$上的执行是指一个有限状态序列$q_0, q_1, q_2, ..., q_n$,其中$q_0$是初始状态,而$q_i = \delta(q_{i-1}, a_i)$, $\forall i.0 < i \leq n$.

定义　接受一个输入串

自动机$M = (Q, \Sigma, \delta, q_0, F)$接受输入串$w = a_1, a_2, ..., a_n \in \Sigma^*$是指存在$w$上的一个执行$q_0, q_1, q_2, ..., q_n$使得$q_n$是$M$的终止状态,即$q_n \in F$. 记作$M \circ\!\!\rightarrow w$.

符号$L(M)$表示M所有能接受的输入串.即$L(M) = \{w \in \Sigma^* \mid M \circ\!\!\rightarrow w\}$.

有限状态自动机

例子：表明上面自动机M_0可以接受下列输入串:

1. 0001
2. 000001111
3. 0101010101
4. 100
5. 0100
6. 110000
7. 0101000000

拒绝如下输入串:

1. 10
2. 101000

语言

定义　语言

设Σ是一个集合,其元素称为字母. Σ^*是Σ的所有有限序列组成的集合. Σ^*的元素称为字, 而其子集称为Σ上的语言.

例子: 设$\Sigma = \{0, 1\}$, 下列集合都是Σ上的语言:

1. $\emptyset$
2. Σ^*
3. $\{000, 1111, 0011, 11000, 11111, 101010101\}$
4. $\{0^n 1^n \mid n > 0\}$
5. $\{0^n 1 \mid n > 0\}$
6. $\{0^n 1 0^m \mid n, m > 0\}$
7. $\{0^n 1^m 0^l \mid n, m, l \geq 0\}$

语言

定义 **识别语言**

设$M = (Q, \Sigma, \delta, q_0, F)$是一自动机,$A$是$\Sigma$上的一语言,若对于$A$中的任何一个元素$w$都有$M \circ\!\!\rightarrow w$,则称自动机$M$可识别语言$A$.即$A \subseteq L(M)$. 记作$M \mapsto A$.

例子:

1. 上面自动机M_0可以识别语言$A = \{w \mid w$至少包含一个1且最后一个1的后面有偶数个零$\}$。
2. 定义自动机$M_1 = (\{q_1, q_2, q_3\}, \delta, \{0, 1\}, q_1, \{q_2\})$,其中$\delta$定义为

	0	1
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_3	q_3

则,M_1可以识别语言$A = \{0^*11^*\}$。

3. 定义自动机$M_2 = (\{q_1, q_2\}, \delta, \{0, 1\}, q_1, \{q_2\})$,其中$\delta$定义为

	0	1
q_1	q_1	q_2
q_2	q_1	q_2

则，M_2可以识别语言$A = \{w \mid w$结束于$1\}$。

正规语言

定义　正规语言

设Σ是一个字母表.Σ上的一个语言A称为正规语言,记作$\models A$, 若存在一个以Σ为字符集的自动机M能识别语言A.

例子：下列语言是$\{0,1\}$上的正规语言：

1. $\{w \mid w$只包含一个$1\}$
2. $\{w \mid w$以0为开头的字$\}$
3. $\{w \mid w$至少包含一个$1\}$.

定义　正规语言类

设Σ是一个字母表.Σ上的所有正规语言组成的类称为Σ的正规语言类, 记作$\mathcal{RL}(\Sigma)$.

注: 正规语言类$\mathcal{RL}$表示所有的字母表Σ上的正规语言类.

正规语言

语言　语言运算

设A和B是语言.定义正规运算并(Union), 链接(Concatenation), 星(Star)运算如下:

- Union: $A \cup B = \{x \mid x \in A \text{ 或} x \in B\}$,
- Concatenation: $A \circ B = \{xy \mid x \in A \text{ 且} y \in B\}$,
- Star: $A^* = \{x_1 x_2 \cdots x_k \mid k \geq 0 \text{ 且每个} x_i \in A\}$.

定理　正规语言类在正规运算下保持不变

设A, B是正规语言,则

1. $A \cup B$是正规语言;
2. $A \circ B$是正规语言;
3. A^*是正规语言.

- $A \cup B$是正规语言.

设计一个能识别语言$A \cup B$的有限自动机M.

不妨设A和B都是Σ上的正规语言,即有Σ上的自动机$M_i = \{Q_i, \Sigma, \delta_i, q_0^i, F_i\}$使得$M_1$识别$A$且$M_2$识别$B$, 即$M_1 |\!\circ\!\!\rightarrow A, M_2 |\!\circ\!\!\rightarrow B$.

构造自动机$M = \{Q_1 \times Q_2, \Sigma, \delta_1 \times \delta_2, (q_0^1, q_0^2), F_1 \times Q_2 \cup Q_1 \times F_2\}$,
其中

$$\begin{aligned} \delta_1 \times \delta_2 : \ (Q_1 \times Q_2) \times \Sigma \ &\longrightarrow \ Q_1 \times Q_2 \\ ((r_1, r_2), \alpha) \ &\longmapsto \ (\delta_1(r_1, \alpha), \delta_2(r_2, \alpha)) \end{aligned}$$

. 下证$M |\!\circ\!\!\rightarrow A \cup B$.

设字$w = \alpha_1\alpha_2 \cdots \alpha_n \in A$,则$M_1 \circ\!\!\rightarrow w$. 从而有一个有限状态序列$q_0^1, q_1^1, \ldots, q_n^1$使得

$$q_0^1 \xrightarrow{\alpha_1} q_1^1 \xrightarrow{\alpha_2} \cdots \xrightarrow{\alpha_{n-1}} q_{n-1}^1 \xrightarrow{\alpha_n} q_n^1$$

且$q_n^1 \in F_1$.

将自动机M_2作用到字w上得到如下M_2的如下序列

$$q_0^2 \xrightarrow{\alpha_1} q_2^1 \xrightarrow{\alpha_2} \cdots \xrightarrow{\alpha_{n-1}} q_{n-1}^2 \xrightarrow{\alpha_n} q_n^2$$

其中且q_n^2未必是M_2的终止状态,即$q_n^2 \in F_2$未必成立.

构造自动机M的有限状态序列:

$$(q_0^1, q_0^2), (q_1^1, q_1^2), \ldots, (q_{n-1}^1, q_{n-1}^2), (q_n^1, q_n^2)$$

这个状态序列具有下面的性质:

$$(q_0^1, q_0^2) \xrightarrow{\alpha_1} (q_1^1, q_2^1) \xrightarrow{\alpha_2} \cdots \xrightarrow{\alpha_{n-1}} (q_{n-1}^1, q_{n-1}^2) \xrightarrow{\alpha_n} (q_n^1, q_n^2)$$

并且$(q_n^1, q_n^2) \in F_1 \times Q_2$是$M$的终止状态.所以自动机$M$接受字$w$, 即$M | \!\circ\!\!\rightarrow w$.

同样可证,若字$w \in B$则$m \circ\!\!\rightarrow w$.

合之证明了$M | \!\circ\!\!\rightarrow A \cup B$.

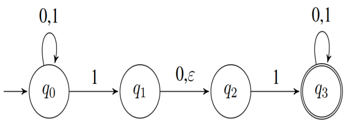
非确定自动机

定义

非确定自动机是一个5元组$(Q, \Sigma, \delta, q_0, F)$, 其中

1. Q是一个状态有限集,其元素称为状态,
2. Σ是一个字母集,其元素称为字母,
3. $\delta : Q \times \Sigma_\varepsilon \longrightarrow \mathcal{P}(Q)$是迁移函数,其中$\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$, $\mathcal{P}(Q)$是Q的幂集,
4. $q_0 \in Q$是初始状态,
5. $F \subseteq Q$是接受/终止状态集.

Back
Close



The nondeterministic inite automata N_1

确定自动机DFA与非确定自动机NFA的区别:

- 对于同一个输入可以转移到多个状态;
- 对于一个输入, 可以没有转移发生;
- 可以输入空串.

定义 非确定自动机计算的形式化定义:

设$N=(Q,\Sigma,\delta,q_0,F)$是一个非确定自动机, w是Σ上的一个串/字.我们称N接受w,记作$N\diamond\!\!\rightarrow w$,若存在$\Sigma_\varepsilon$的有限序列$y_1,y_2,\ldots,y_m$使得$w=y_1y_2\cdots y_m$以及$Q$的一个有限序列$r_0,r_1,\cdots,r_m$使得下面三条成立:

1. $r_0=q_0$,
2. $r_{i+1}\in\delta(r_i,y_{i+1})$, 对于$i=0,1,\ldots,m-1$,
3. $r_m\in F$.

注: $\varepsilon y=y\varepsilon=y$对于所有的$y\in\Sigma$.

非确定自动机

定义　识别

设$N=(Q,\Sigma,\delta,q_0,F)$是一个非确定自动机, A是Σ的一个语言.若A中的每一个字w都能被非确定自动机N识别,则称非确定自动机N 识别语言A, 记作$N|\!\!\diamond\!\!\!\longrightarrow A$.

定理　等价性

非确定自动机与确定自动机在表达能力上是等价的,即非确定自动机所识别的语言类与确定自动机所识别的语言类是一致的.

- 确定自动机能识别的语言类是非确定自动机所识别语言类的子类.

因为确定自动机一定是非确定自动机

- 非确定自动机能识别的语言类是确定自动机所识别语言类的子类

设$N=(Q,\Sigma,\delta,q_0,F)$是一非确定自动机,且识别语言$A$,即$N|\!\!\diamond\!\!\!\longrightarrow A$.我们将构造一个也能识别$A$的确定自动机$M=(Q',\Sigma,\delta',q_0',F')$,即$M|\!\!\diamond\!\!\!\longrightarrow A$.

1. $Q'=\mathcal{P}(Q)$

2. $\delta' : Q' \times \Sigma \longrightarrow Q'$

   $\delta'(R, a) = \{q \in Q \mid q \in E(\delta(r, a))$ 对某个$r \in R\}$.

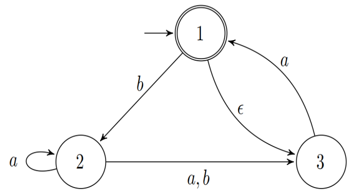
   其中$E(R) = \{q \mid$ 从R仅通过零个箭头或ε箭头达到q $\}$. 因此,$R \subseteq E(R)$.

3. $q_0' = E(\{q_0\})$

4. $F' = \{R \in Q' \mid R$ 至少包含N的一个终止状态$\}$.

可以证明: 确定自动机$M \models\!\!\!\rightarrow A$.

例子:非确定自动机$N_4 = (Q, \{a, b\}, \delta, 1, \{1\})$, 构造等价的DFA.



The NFA N_4

第一步:确定D的状态.

$$\{\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}.$$

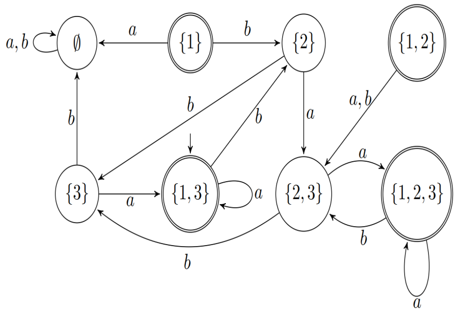
第二步:确定转移函数. 例如:

$$\begin{array}{lcl}
\{2\} & \xrightarrow{a} & \{2,3\} \\
\{2\} & \xrightarrow{b} & \{3\} \\
\{1\} & \xrightarrow{a} & \emptyset \\
\{3\} & \xrightarrow{a} & \{1,3\} \\
\{2,3\} & \xrightarrow{a} & \{1,2,3\} \\
\{2,3\} & \xrightarrow{b} & \{3\}
\end{array}$$

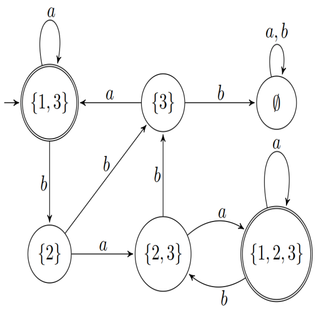
第二步:确定开始状态和接受状态.

开始状态: $E(\{1\}) = \{1,3\}$;

接受状态:$\{\{1\}, \{1,2\}, \{1,3\}, \{1,2,3\}\}$.



A DFA D is equivalent to the NFA N_4



DFA D after removing unnecessary states

非确定自动机

推论

一个语言是正规的当且仅当某个非确定有限自动机能够识别它.

定理

正规语言在语言的正规运算下保持不变. 设A, B是Σ上的正规语言,再设N_A, N_B是分别识别A, B的非确定自动机.

- 在并运算下保持不变.

构造一个非确定自动机N_1,能识别语言$A \cup B$.

$N_1 = (Q_A \cup Q_B \cup \{q_0\}, \Sigma, \delta', q_0, F_A \cup F_B)$,其中
$\delta'(q_0, \varepsilon) = q_0^A$或$q_0^B$, 而$\delta'(q^A, a) = \delta^A(q^A, a)$
$\delta'(q^B, a) = \delta^B(q^B, a)$.

- 在链接运算下保持不变.

构造一个非确定自动机N_2,能识别语言$A \circ B$.

$N_2 = (Q_A \cup Q_B, \Sigma, \delta', q_0^A, F_B)$,其中
$\delta'(q, \varepsilon) = q_0^B$, 若$q \in F_A$.
$\delta'(q, a) = \delta^A(q, a)$, 若$q \in Q^A, \notin F_A$.
$\delta'(q, a) = \delta^B(q, a)$, 若$q \in Q^B$.

- 在星运算下保持不变.

构造一个非确定自动机N_3,能识别语言A^*.

$$N_3 = (\{q_0\} \cup Q^A, \Sigma, \delta', q_0, \{q_0\} \cup F_A)$$

$$\delta'(q,a) = \begin{cases} \delta^A(q,a) & q \in Q^A \wedge a \notin F_A \\ \delta^A(q,a) & q \in F_A \wedge a \neq \varepsilon \\ \delta^A(q,a) \cup \{q_0^A\} & q \in F_A \wedge a = \varepsilon \\ q_0^A & q = q_0 \wedge a = \varepsilon \\ \emptyset & q = q_0 \wedge a \neq \varepsilon. \end{cases}$$

作业

1. 构造确定自动机分别能识别下面的语言,其中字母集为$\{0,1\}$
   (a) $\{w \mid w$ 由1开头,由一个0结尾$\}$
   (b) $\{w \mid w$ 至少包含3个1 $\}$
   (c) $\{w \mid w$在奇数位置是1$\}$
   (d) $\{w \mid w$的长度至多为5$\}$
2. 构造非确定自动机分别能识别下面的语言,其中字母集为$\{0,1\}$
   (a) $\{w \mid w$以00结尾$\}$并且仅有3个状态
   (b) $\{0\}$仅有2个状态
3. 证明非确定自动机可以等价地转换为确定自动机.

非正规语言

例子：设$\Sigma = \{0, 1\}$, Σ上的下列语言不是正规语言

1. $C = \{w \mid w$ 拥有相同个数的0和1$\}$.
2. $B = \{0^n 1^n \mid n \geq 0\}$.
3. $F = \{ww \mid w \in \{0, 1\}^*\}$.

但是: $D = \{w \mid w$ 拥有相同个数的子串01和10$\}$是正规的.

定理 泵引理(Pumping Lemma)

设A是一个正规语言,则存在一个数p(泵的长度)使得: 若s是A的长度至少是p的任意字, 那么s一定可分解成三段: $s = xyz$并且满足下面的三条:

1. 对于任何$i \geq 0$都有$xy^iz \in A$,
2. $|y| > 0$,
3. $|xy| \leq p$.

其中$|x|$表示串x的长度,y^i表示将y的i次拷贝链接在一起,而y^0是空串ε. p一般是能识别语言A的自动机的状态个数.

证明

设$M=(Q,\Sigma,\delta,q_1,F)$是识别语言$A$的DFA且$p$是$M$的状态数.

假定$s=s_1s_2\cdots s_n$ 是A中长度为n的串且$n\geq p$. 设$r_1,r_2,\cdots r_{n+1}$是输入s所产生的状态序列, 因此$r_{i+1}=\delta(r_i,s_i)(1\leq i\leq n)$. 该序列的长度是$n+1$,其大于或等于$p+1$. 这样该序列的前$p+1$个元素中必定有两个元素相同.进一步, 记这两个相同的元素第一次出现为r_j,第二次出现为r_l, 则有$l\leq p+1$.

令$x=s_1\cdots s_{j-1},y=s_j\cdots s_{l-1},z=s_l\cdots s_n$.则

(1)因为x产生的状态是从r_1到r_j;y产生的状态是从r_j到r_j;z产生的状态是从r_j到r_{n+1}, 而r_{n+1}是一接受状态,所以M必定接受$xy^iz(i\geq 0)$, 即$xy^iz\in A$;

(2)由于$j\neq l$, 所以$|y|>0$;

(3)因为$l\leq p+1$, 所以$|xy|\leq p$.

例子:利用泵引理证明$B=\{0^n1^n \mid n\geq 0\}$不是正规语言.

证明:假设B是正规的. 选择$s=0^p1^p$, 其中p是泵的长度. 因s是B中的字且长度大于p, 据泵引理s可以分成三部分$s=xyz$, 对于任意的$i\geq 0$, $xy^iz\in B$,考虑三种情况:

(1) y只含有0
此时$xyyz$含有0的个数比1的个数多,故不是B中的元素, 矛盾!

(2) y只含有1
此时$xyyz$含有1的个数比0的个数多,故也不是B中的元素, 矛盾!

(3) y含有0与1
此时$xyyz$可能含有相同的0和1, 但是顺序出错,也不是B中的元素,矛盾!

下推自动机

定义

一个下推自动机是一个6元组$(Q, \Sigma, \Gamma, \delta, q_0, F)$，其中$Q, \Sigma, \Gamma, F$都是有限集，并且

1. Q是状态集，
2. Σ是输入字符集，
3. Γ是栈符集，
4. $\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \longrightarrow \mathcal{P}(Q \times \Gamma_\varepsilon)$ 是转移函数，
5. $q_0 \in Q$是始状态，
6. $F \subseteq Q$是终状态集。

Back

Close

例子: 下推自动机M_1:

1. $Q = \{q_1, q_2, q_3, q_4\}$,
2. $\Sigma = \{0, 1\}$,
3. $\Gamma = \{0, \$\}$,
4. $F = \{q_1, q_4\}$,
5. δ定义如下:

$$\begin{aligned}
&\delta(q_1, \varepsilon, \varepsilon) = \{(q_2, \$)\}\\
&\delta(q_2, 0, \varepsilon) = \{(q_2, 0)\}\\
&\delta(q_2, 1, 0) = \{(q_3, \varepsilon)\}\\
&\delta(q_3, 1, 0) = \{(q_3, \varepsilon)\}\\
&\delta(q_3, \varepsilon, \$) = \{(q_4, \varepsilon)\}
\end{aligned}$$

下推自动机

定义

一个下推自动机(Pushdown Automata)$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$的计算定义如下：$M$可以接受输入$w$，若$w = w_1 w_2 \cdots w_m$ (其中$w_i \in \Sigma_\varepsilon$)且存在状态序列$r_0, r_1, \ldots, r_m$和串序列$s_0, s_1, \ldots, s_m \in \Gamma^*$ 满足下面的三条:

1. $r_0 = q_0, s_0 = \varepsilon$,
2. 对于每个$i = 0, 1, \ldots, m-1$，有$(r_{i+1}, b) \in \delta(r_i, w_{i+1}, a)$，其中$s_i = at_1$, $s_{i+1} = bt_2$ $(a, b \in \Gamma_\varepsilon, t_i \in \Gamma^*)$,
3. $r_m \in F$.

例子: 前面下推自动机M_1 可以接受语言$\{0^n 1^n \mid n \geq 0\}$.
M_1识别字$0^3 1^3$.

$$\begin{aligned}(q_1, \varepsilon) &\xrightarrow{\varepsilon} (q_2, \$) \xrightarrow{0} (q_2, 0\$) \xrightarrow{0} (q_2, 00\$) \xrightarrow{0} (q_2, 000\$) \\ &\xrightarrow{1} (q_3, 00\$) \xrightarrow{1} (q_3, 0\$) \xrightarrow{1} (q_3, \$) \xrightarrow{\varepsilon} (q_4, \varepsilon).\end{aligned}$$

上下文无关语言

定义　上下文无关语法Context-free grammar CFG

一个上下文无关语法是一个4元组(V, Σ, R, S)，其中

1. V是一个有限集，其元素称为变量，
2. Σ是一个与V不相交的有限集,其元素成为终结符，
3. R是一个有限集，其元素称为规则,而每一个规则包含一个变量和一个由变量以及终结符组成的串，
4. $S \in V$是始变量.

例子: 上下文无关语法$G_3 = (\{S\}, \{a, b\}, R, S)$,其中规则$R$定义为

$$S \to aSb \mid SS \mid \varepsilon$$

例子：上下文无关语法$(\{S\}, \{0, 1\}, R, S)$，其中R定义为

$$S \to 0S1 \mid \varepsilon$$

定义

上下文无关文法$G=(V,\Sigma,R,S)$. 设u,v,w是$V\cup\Sigma$上的串,若$A\to w\in R$, 则我们称uAv产生了uwv ,记作$uAv\Rightarrow uwv$.

称u导出v,记作$u\overset{*}{\Rightarrow}v$, 若$u=v$或存在序列$u_1,u_2,\cdots,u_k$使得$u\Rightarrow u_1\Rightarrow u_2\Rightarrow\ldots\Rightarrow u_k\Rightarrow v$. 即$\overset{*}{\Rightarrow}$是$\Rightarrow$的自反传递闭包.

G生成了语言是$\{w\in\Sigma^*\mid S\overset{*}{\Rightarrow}w\}$.

上下文无关语言

定义　上下文无关语言

上下文无关文法生成的语言称为上下文无关语言.

例子：上下文无关语法$(\{S\},\{0,1\},R,S)$，其中R定义为$S \to 0S1 \mid \varepsilon$ 生成语言$\{0^n1^n \mid n \geq 0\}$.

定理

一个语言是上下文无关语言当且仅当它可被一个下推自动机识别。

设上下文无关文法$G=(V,\Sigma,R,S)$生成语言A。构造一个下推自动机$P=(Q,\Sigma,\delta,q_{start},\{q_{accept}\})$ 如下:

1. $Q=\{q_{start},q_{\$},q_{loop},q_{accept}\}$
2. $\delta(q_{start},\varepsilon,\varepsilon)=\{(q_{\$},\$)\}$,
3. $\delta(q_{\$},\varepsilon,\varepsilon)=\{(q_{loop},S)\}$
4. $\delta(q_{loop},\varepsilon,A)=\{(q_{loop},w)\ |$其中$A\to w\in R\}$.
5. $\delta(q_{loop},a,a)=\{(q_{loop},\varepsilon)\}$
6. $\delta(q_{loop},\varepsilon,\$)=\{(q_{accept},\varepsilon)\}$.

则P可以识别语言A.

再设下推自动机$P=(Q,\Sigma,\delta,q_0,q_{accept})$识别语言$A$。现构造一个上下文无关文法$G=(V,\Sigma,R,S)$如下:

- $V=\{A_{pq}\mid p,q\in Q\}$.
- R定义如下:
  1. $A_{pq}\to aA_{rs}b$若$(r,t)\in\delta(p,a,\varepsilon)$ 且$(q,\varepsilon)\in\delta(s,b,t)$,其中$p,q,r,s\in Q,t\in\Gamma,a,b\in\Sigma_\varepsilon$,
  2. $A_{pq}\to A_{pr}A_{rq},p,q,r\in Q$,
  3. $A_{pp}\to\varepsilon,p\in Q$.
- $S=A_{q_0q_{accept}}$.

上下文无关语言

定理

上下文无关语言在语言的并运算、链接运算以及Kleene星运算是封闭的。

证明: 设$G_1 = (V_1, \Sigma_1, R_1, S_1)$ 及$G_2 = (V_2, \Sigma_2, R_2, S_2)$是上下文无关文法.

- 并运算: 构造$G = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, R, S)$其中$R = R_1 \cup R_2 \cup \{S \to S_1, S \to S_2\}$. 则$\mathcal{L}(G) = \mathcal{L}(G_1) \cup \mathcal{L}(G_2)$.
- 链接运算: 构造$G = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, R_1 \cup R_2 \cup \{S \to S_1S_2\}, S)$.则$\mathcal{L}(G) = \mathcal{L}(G_1)\mathcal{L}(G_2)$.
- Kleene星运算: $G = (V_1 \cup \{S\}, \Sigma_1, R_1 \cup \{S \to \varepsilon, S \to SS_1\}, S)$. 则$\mathcal{L}(G) = \mathcal{L}(G_1)^*$.

作业

1. 证明: 上下文无关语言在语言的并运算、链接运算以及Kleene星运算是封闭的.
2. 构造一个上下文无关文法G和一个下推自动机P 都生成$\{0,1\}$上的语言$\{w \mid w$ 开始与结尾都是同一个字符$\}$.

非上下文无关语言

例子: $\{a^ib^jc^k \mid 0 \le i < j \le k\}$.

$\{a^nb^nc^n \mid n \ge 0\}$.

$\{a^n \mid n \ge 1\ n$ 是素数$\}$.

定理 (泵引理)

设A是一个上下文无关语言，则存在一个数p(称为泵的长度)使得A任何长度大于等于p字w 都能分成5份$w = uvxyz$满足下面三条:

1. 对于每一个i都有$uv^ixy^iz \in A$,
2. $|vy| > 0$,
3. $|vxy| \le p$.

图灵自动机

定义　图灵机

Turing机是一个7元组$(Q,\Sigma,\Gamma,\delta,q_0,q_{accept},q_{reject})$,其中

1. Q 是一个状态集,
2. Σ是输入字母集，但不包括空符$\sqcup$,
3. Γ是带上符,其中$\{\sqcup,\triangleright\}\subseteq\Gamma,\Sigma\subseteq\Gamma$,
4. $\delta: Q\times\Gamma\longrightarrow Q\times\Gamma\times\{L,R\}$是转移函数,
5. $q_0\in Q$是始状态符,
6. $q_{accept}\in Q$是接受符,
7. $q_{reject}\in Q$是拒绝符，其中$q_{accept}\neq q_{reject}$.

图灵机

例子: TM $M_0 = (\{q_0, q_1, q_2, q_3, q_{accept}, q_{reject}\}, \{0,1\}, \{0,1,X,Y,\triangleright,\sqcup\}, \delta, q_0, q_{accept}, q_{rejct})$

state	Symbol $\triangleright$	0	1	X	Y	$\sqcup$
q_0	$(q_1, \triangleright, R)$	-	-	-		
q_1		(q_2, X, R)	-	-	(q_4, Y, R)	-
q_2		$(q_2, 0, R)$	(q_3, Y, L)	-	(q_2, Y, R)	-
q_3		$(q_3, 0, L)$	-	(q_1, X, R)	(q_3, Y, L)	-
q_4		-	-	-	(q_4, Y, R)	$(q_a, \sqcup, R)$

例子: TM $M_1 = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$:

- $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_{accet}, q_{reject}\}$,
- $\Sigma = \{0\}$,
- $\Gamma = \{0, X, \triangleright, \sqcup\}$,
- δ定义见表,
- q_1是始状态,q_{accept}是接受状态,q_{reject}是拒绝状态.

state	Symbol $\triangleright$	0	X	$\sqcup$
q_0	$(q_1, \triangleright, R)$			
q_1		$(q_2, \sqcup, R)$	(q_r, X, R)	$(q_r, \sqcup, R)$
q_2		(q_3, X, R)	(q_2, X, R)	$(q_a, \sqcup, R)$
q_3		$(q_4, \sqcup, R)$	(q_3, X, R)	$(q_5, \sqcup, L)$
q_4		(q_3, X, R)	(q_4, X, R)	$(q_r, \sqcup, R)$
q_5		$(q_5, 0, L)$	(q_5, X, L)	$(q_2, \sqcup, R)$

Back
Close

图灵机–格局

定义　格局-Configuration

图灵机的一个格局是一个由当前状态、当前带上字符，以及当前指针头位置组成的3元组.

用符号uqv表示一个格局, 其中u, v是Γ上的串,uv是当前带上的字符串, q是状态符,指针头是v的左边第一个字符.如:$1011(\underline{q_7 0})1111$.

1. 始格局(Start Configuration): $q_0 \triangleright w \sqcup$,其中$q_0$是始状态, 指针头位于$\triangleright$的左边的第一个字符.
2. 接受格局(Accepting Configuration): 格局中的状态是q_{accept},
3. 拒绝格局(Rejecting Configuration): 格局中的状态是q_{reject},
4. 停止格局(Halting Configuration): 接受格局和拒绝格局统称为停止格局.

格局–例子

1. 字中至少有一个a
   - 初始格局：$q_0 \rhd bbbbbaaabbb\sqcup$
   - 接受格局：$\rhd bbbbbbaq_{accept}babaabaab\sqcup$.
   - 拒绝格局：$\rhd bbbbbbb \sqcup q_{reject}$。
2. 字是仅由2^n个数0组成的串
   - 初始格局$q_0 \rhd 0000000000\sqcup$
   - 接受格局$\rhd 00000000 \sqcup q_a$
   - 拒绝格局$\rhd 000000000000 \sqcup q_r$

Back
Close

图灵机–格局演算

定义　格局演算

称格局C_1产出格局C_2,记作$C_1 \vdash C_2$,若C_1和C_2是下列两种情形:

1. $C_1 = u(\underline{aq_ib})v$,$C_2 = u(\underline{q_jac})v$，且$\delta(q_i, b) = (q_j, c, L)$,
2. $C_1 = u(\underline{aq_ib})v$,$C_2 = u(\underline{acq_j})v$，且$\delta(q_i, b) = (q_j, c, R)$.

定义　接受输入

TM接受输入w: 若存在一个格局序列$C_1, C_2, \ldots, C_k$使得

1. C_1是始格局,
2. $C_i \vdash C_{i+1}$, $i = 1, \ldots k-1$,
3. C_k是接受格局.

图灵机可识别语言

定义　接受/识别

图灵机M接受语言L或M识别语言L,若L中的每一个元素都被M接受。

定义　图灵识别语言

一个语言称为图灵可识别的,若有一个图灵机可以识别它.

例子：TM可识别语言$L = \{w \in \{a,b\}^* \mid w$至少包含一个$a\}$.

- 状态集$Q = \{q_0, q_1, q_a, q_r\}$, 其中$q_0$是初始状态，$q_a$是接受状态，$q_r$是拒绝状态,
- 输入字母集$\Sigma = \{a, b\}$.
- 带上符号集$\Gamma = \{a, b, \triangleright, \sqcup\}$,
- 转移函数δ定义如下表:

state	Symbol $\triangleright$	a	b	$\sqcup$
q_0	$(q_1, \triangleright, R)$			
q_1		(q_a, a, R)	(q_1, b, R)	

例子

TM $M_0 = (\{q_0, q_1, q_2, q_3, q_4 q_a, q_r\}, \{0,1\}, \{0,1,X,Y,\triangleright,\sqcup\}, \delta, q_0, q_{accept}, q_{rejct})$

state	$\triangleright$	0	1	X	Y	$\sqcup$
	Symbol					
q_0	$(q_1, \triangleright, R)$					
q_1		(q_2, X, R)	-	-	(q_4, Y, R)	-
q_2		$(q_2, 0, R)$	(q_3, Y, L)	-	(q_2, Y, R)	-
q_3		$(q_3, 0, L)$	-	(q_1, X, R)	(q_3, Y, L)	-
q_4		-	-	-	(q_4, Y, R)	$(q_a, \sqcup, R)$

图灵机M_0可识别语言: $\{0^n 1^n \mid n \geq 1\}$.

$M_0 =$ "对输入串w":

1. 扫描带子，若发现1右边有0则拒绝,
2. 若0和1都在带子上，重复下面两条:
3. 扫描带子，删除一个0和一个1,
4. 若在所有1删除后还有0剩下，或在所有0都删除后还有1剩下，则拒绝, 若没有0和1剩下，则接受.

例子

- 设计TM能识别语言 $B = \{w\sharp w \mid w \in \{0,1\}^*\}$

设 $\Sigma = \{0, 1, \sharp\}$, $\Gamma = \{0, 1, \sharp, X, \sqcup, \triangleright\}$, 转移函数 δ 定义如下表：

	Symbol					
state	$\triangleright$	0	1	$\sharp$	X	$\sqcup$
q_0	$(q_1, \triangleright, R)$					
q_1		(q_2, X, R)	(q_3, X, R)	$(q_8, \sharp, R)$		
q_2		$(q_2, 0, R)$	$(q_2, 1, R)$	$(q_4, \sharp, R)$		
q_3		$(q_3, 0, R)$	$(q_3, 1, R)$	$(q_5, \sharp, R)$		
q_4		(q_6, X, L)			(q_4, X, R)	
q_5			(q_6, X, L)		(q_5, X, L)	
q_6		$(q_6, 0, L)$	$(q_6, 1, L)$	(q_7, X, L)	(q_6, X, L)	
q_7		$(q_7, 0, L)$	$(q_7, 1, l)$		(q_1, X, R)	
q_8					(q_8, X, R)	$(q_a, \sqcup, R)$

Back
Close

设计图灵机–例子

- 设计TM能识别语言$B = \{a^n b^n c^n \mid n \geq 0\}$

设计策略：对于输入字$a^n b^n c^n$，在带子上构造串为$\triangleright a^n b^n c^n \sqcup$,

1. 从最左边符号$\sharp$开始，发现第一个a变成d，然后右移找b
2. 找到第一个b,将b变成d,然后右移
3. 找到第一个c，将c变成d，则左移直到找到$\sharp$,然后重复上述过程。

设计状态q_1, q_2, q_3, q_4, q_5,其功能如下：

- 状态q_1: 负责把a变成d.

  从左往右找到第一个a将a变成d,并转移给状态q_2，若遇到d则直接右移，之后遇到了b 则表明b的个数比a多，因此拒绝，同样，若遇到c表明c比a的个数多，同样拒绝。

- 状态q_2: 负责把b变成d.

  从左往右，遇到a或d直接右移，遇到第一个b将b变成d,并转移给状态q_3，若遇到d则直接右移，之后遇到了c则表明c的个数比b多，因此拒绝

- 状态q_3: 负责将c变成d.

  从左往右，遇到b或d直接右移, 遇到第一个c将c变成d,并转移给状态q_4,

- 状态q_4: 负责识别能否接受。

  若遇到$\sqcup$则接受输入，若遇到c则左移并交给状态q_5

- 状态q_5：负责左移直到最左端.

  遇到d, b, a左移，直到$\rhd$,遇到$\rhd$ 转移给状态q_1.

例子

- 设计TM能识别语言$B = \{a^n b^n c^n \mid n \geq 0\}$

设计策略：对于输入字$a^n b^n c^n$，在带子上构造串为$\triangleright a^n b^n c^n \sqcup$,

state	Symbol $\triangleright$	a	b	c	d	$\sqcup$
q_0	$(q_1, \triangleright, R)$					
q_1		(q_2, d, R)	$(q_r, \sqcup, R)$	$(q_r, \sqcup, R)$	(q_1, d, R)	
q_2		(q_2, a, R)	(q_3, d, R)	$(q_r, \sqcup, R)$	(q_2, a, R)	
q_3			(q_3, b, R)	(q_4, d, R)	(q_3, d, R)	
q_4				(q_5, c, L)		$(q_a, \sqcup, R)$
q_5	$(q_1, \triangleright, R)$	(q_5, a, L)	(q_5, b, L)		(q_5, d, L)	

例子

- 构造TM计算$n+1$函数

设计策略：将n转化成$0,1$代码，依据$0+1=1$以及$1+1=10$原则进行设计.转移函数δ如下：

state	Symbol $\triangleright$	0	1	$\sqcup$
q_0	$(q_1, \triangleright, R)$			
q_1		$(q_1, 0, R)$	$(q_1, 1, R)$	$(q_2, \sqcup, L)$
q_2		$(q_3, 1, R)$	$(q_2, 0, L)$	
q_3		$(q_3, 0, R)$	$(q_3, 1, R)$	$(q_a, \sqcup, R)$

Back

Close

图灵机变异

定义 多带图灵机

$$\begin{aligned} \delta:\ & Q\times\Gamma^k \longrightarrow Q\times\Gamma^k\times\{L,R,S\}^k \\ & (q_i,a_1,a_2,\ldots,a_k)\longmapsto(q_j,b_1,b_2,\ldots,b_k,L,R,\ldots,L) \end{aligned}$$

即：

$\delta(q_i,a_1,a_2,\ldots,a_k)=(q_j,b_1,b_2,\ldots,b_k,L,R,\ldots,L).$

定义 非确定图灵机

$$\delta: Q\times\Gamma\longrightarrow\mathcal{P}(Q\times\Gamma\times\{L,R\}).$$

图灵机变异

定理　等价性

1. 每一个多带图灵机都有一个等价的单带图灵机.
2. 每一个非确定图灵机都有一个等价的确定图灵机.

可判定语言

可判定语言

定义　判定器

一个图灵机称为判定器，若它总是做出接受还是拒绝一个输入，即它总是从始格局停止于一个接受格局或拒绝格局.

定义　图灵可判定语言

图灵机可判定的语言称为图灵可判定或可判定。称Σ上的语言L是图灵可判定的，若存在一个图灵机M使得对Σ的任何字$\omega \in \Sigma^*$都有，若$\omega \in L$则M接受ω否则拒绝ω。

定理

每一个图灵可判定语言都是图灵可识别的语言.

例子

下面TM可判定语言$L = \{w \in \{a,b\}^* \mid w$至少包含一个$a\}$.

state	Symbol $\triangleright$	a	b	$\sqcup$
q_0	$(q_1, \triangleright, R)$			
q_1		(q_a, a, R)	(q_1, b, R)	$(q_r, \sqcup, R)$

Back
Close

例子

TM $M_1 = (Q, \Sigma, \Gamma, \delta, q_1, q_{accept}, q_{reject})$:

- $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_{accet}, q_{reject}\}$,
- $\Sigma = \{0\}$,
- $\Gamma = \{0, X, \triangleright, \sqcup\}$,
- δ定义见图,
- q_1是始状态,q_{accept}是接受状态,q_{reject}是拒绝状态.

图灵判定语言: $\{0^{2^n} \mid n \geq 0\}$.

state	Symbol $\triangleright$	0	X	$\sqcup$
$(q_0$	$(q_1, \triangleright, R)$			
q_1		$(q_2, \sqcup, R)$	(q_r, X, R)	$(q_r, \sqcup, R)$
q_2		(q_3, X, R)	(q_2, X, R)	$(q_a, \sqcup, R)$
q_3		$(q_4, 0, R)$	(q_3, X, R)	$(q_5, \sqcup, L)$
q_4		(q_3, X, R)	(q_4, X, R)	$(q_r, \sqcup, R)$
q_5		$(q_5, 0, L)$	(q_5, X, L)	$(q_2, \sqcup, R)$

可判定语言

定义

设$A_{\mathsf{DFA}} = \{< B, w >|\ B$是一个确定自动机且可以接受输入串$w\}$.
设$E_{\mathsf{DFA}} = \{< A >|\ A$是一个确定自动机且$\mathcal{L}(A) = \emptyset\}$.
设$EQ_{\mathsf{DFA}} = \{< A, B >|\ A, B$是确定自动机且$\mathcal{L}(A) = \mathcal{L}(B)\}$.
设$A_{\mathsf{CFG}} = \{< G, w >|\ G$是一个上下文无关文法且可以生成串$w\}$.
设$E_{\mathsf{CFG}} = \{< G >|\ G$是一个上下文无关文法且$\mathcal{L}(G) = \emptyset\}$.
设$A_{\mathsf{TM}} = \{< M, w >|\ M$是一个图灵机且可以接受输入串$w\}$.

可判定语言

定理

$A_{\mathsf{DFA}}, E_{\mathsf{DFA}}, EQ_{\mathsf{DFA}}, A_{\mathsf{CFG}}, E_{\mathsf{CFG}}$都是可判定语言. 但$A_{\mathsf{TM}}$是不可判定的.

证明A_{DFA}是图灵可判定的. 我们需要构造一个图灵机M,它能判定A_{DFA}.

M定义为: 输入$< B, w >$其中B是一个确定自动机,w是一个串,

1. 模拟B在w上的行为,
2. 若模拟在接受状态上结束,则接受;若模拟在非接受状态上结束,则拒绝.

- M 首先检查$< B, w >$是否是一个合法的输入, 即$< B, w >$是表示一个自动机B和B一个输入, 若不是合法则M拒绝, 此时$B = (Q, \Sigma, \delta, q_0, F)$,$w$是$\Sigma$上的一个有限串.
- M模拟B在w上轨迹: B的现在状态, B现在位置,M的方向总是向右.
  - 初始B的现在状态是q_0,而B的现在位置是w的最左边字符. 然后状态和位置随着δ的变化而改变,方向总是向右,

- 若B在w是最右边的一个字符时状态为终止状态,则M接受串w,否则拒绝w。

注：A_{TM}是图灵可识别的.

正规语言$\longrightarrow$ 上下文无关语言$\longrightarrow$ 图灵可判定语言$\longrightarrow$ 图灵可识别语言.

丘奇-图灵命题

- 1900年: 德国数学家希尔伯特(Davis Hilbert)在巴黎举行的数学家大会上提出的第10个问题：设计一个算法测试多项式是否有整数根.
- 1936: 美国数学家Alonzo Church 使用λ演算来研究Hibert的第10个问题
- 1936: 英国数学家Alan Turing 使用图灵机研究Hilbert的第10个问题
- 1970: Yuri Matijasevic证明检查多项式是否有整数根的算法是不存在的.
- 1952: 美国数学家斯蒂芬·科尔·克莱尼(Stephen Cole Kleene)和逻辑学家J.B. Rosser一起定义了一类函数，这种函数的值可使用递归方法计算,称为递归函数
- 邱奇-图灵论题：如果某个算法是可行的，那这个算法可以被图灵机和λ演算（以及递归函数）表达。

时间计算复杂性

例子: 图灵判定语言$A = \{0^n1^n \mid n \geq 0\}$. 设图灵机$M_1$可以判定它, M_1定义如下：

$M_1 =$ ”对输入串w”:

1. 扫描带子，若发现1右边有0则拒绝,
2. 若0和1都在带子上，重复下面两条:
3. 扫描带子，删除一个0和一个1,
4. 若在所有1删除后还有0剩下，或在所有0都删除后还有1剩下，则拒绝, 若没有0和1剩下，则接受.

时间计算复杂性

分析判定A的图灵机M_1的算法，确定它需要多少时间.分四步分析：

第一步：扫描带子是验证输入串是否属于形式0^*1^*，从左到右扫描一遍，然后指针回到最左边,共用了$2n$步, 是$O(n)$,

第二、三步：扫描带子，每一次扫描最多需要$O(n)$时间，由于每次扫描要删除一个0和一个1，所以最多扫描$n/2$次,共需要$(n/2)O(n) = O(n^2)$时间,

第四步: 一次扫描就可以完成，因此最多时间是$O(n)$,

这样，M_1在长度为n的输入串上共需要时间是$O(n) + O(n^2) + O(n) = O(n^2)$.

时间计算复杂性

定义 $f(n)$时间图灵机

设M是一个确定图灵机，且M对所有的输入串都能停机. M的时间复杂性或运行时间是函数$f:\mathcal{N}\longrightarrow\mathcal{N}$, 其中$f(n)$是$M$在长度为$n$的任何输入上所使用的最大步数.

若$f(n)$是图灵机M的时间复杂函数,则称M是$f(n)$时间图灵机.

时间计算复杂性

例子：判定语言$\{0^k1^k \mid k \geq 0\}$的图灵机M_1是$O(n^2)$时间图灵机.
问题：能否设计一个图灵机可以更快地判定语言A?
设计图灵机M_2判定语言$A = \{0^n1^n \mid n \geq 0\}$，$M_2$定义如下：
$M_2 =$ ”对输入串w”:

1. 扫描带子，若发现1右边有0则拒绝,
2. 只要带子上有0和1，重复下面三条:
3. 扫描带子，检查带子上0和1的总数是否是奇数？若是则拒绝,
4. 再次扫描带子，从第一个0开始每隔一个0删除一个0，从第一个1开始每隔一个1删除一个1,
5. 若带子上没有0和1剩下则接受,否则拒绝.

M_2运行的时间是$O(n) + O(n \log n) = O(n \log n)$.

时间计算复杂性

定义　时间复杂类

设$t:\mathcal{N}\longrightarrow\mathcal{R}^+$是一个函数. 时间复杂类$\mathsf{TIME}(t(n))$是一个所有语言族，其中每一个语言都可以被$O(t(n))$时间图灵机判定.

定义　　P类

P类语言是指一类语言，其中每一个语言都可以被多项式时间确定单带图灵机判定，即

$$\mathsf{P} = \bigcup_k \mathsf{TIME}(n^k)$$

例子：$PATH$问题$\in$ P，其中

$PATH = \{< G, s, t >|\ G$是一个有向图并且从s到t有一条有向路$\}$.

Back
Close

时间复杂性

定义 ($f(n)$时间非确定图灵机)

设N是一个非确定图灵机（判定器），函数$f: \mathcal{N} \longrightarrow \mathcal{N}$称为$N$的运行时间函数, 其中$f(n)$是$N$在长度为$n$的任何输入上所有分支使用的最大步数.

定义

$\mathsf{NTIME}(t(n)) = \{L \mid L$是一个被$O(t(n))$时间非确定图灵机判定的语言$\}$.

定理

设$t(n): \mathcal{N} \longrightarrow \mathcal{N}$是函数且$t(n) \geq n$. 则每一个$t(n)$时间非确定单带图灵机都有一个等价的$2^{O(t(n))}$时间确定单带图灵机.

定义 NP类

NP类语言是指一类语言，其中每一个语言都可以被某个非确定的多项式时间图灵机判定。即:

$$\mathsf{NP} = \bigcup_k \mathsf{NTIMES}(n^k).$$

例子:

$HAMPATH = \{< G, s, t >|\ G$是一个有向图且从$s$到$t$有Hamiltonian路$\}$.

定理

$$HAMPATH \in \mathsf{NP}.$$

问题: $HAMPATH \in \mathsf{P}$?

定理　Cook-Levin 定理

设$SAT = \{< \phi >| \ \phi$ 是可满足的布尔公式$\}$. 则

$$SAT \in \mathsf{P}\text{当且仅当}\mathsf{P} = \mathsf{NP}.$$

定义　多项式时间可计算函数

称函数$f:\Sigma^* \longrightarrow \Sigma^*$是多项式时间可计算函数，若存在一个多项式时间图灵机M使得对于任何输入w, 在带子上,M都停止在$f(w)$上.

定义　多项式时间规约

称语言A多项式时间映射规约,简称多项式时间规约，到语言B, 记作$A \leq_P B$，若有一个多项式时间可计算函数$f:\Sigma^* \longrightarrow \Sigma^*$使得

$$w \in A \text{当且仅当} f(w) \in B.$$

函数f称为A到B的多项式时间规约.

Back
Close

NP完备

定义　NP完备

语言B称为NP完备,若$B \in \mathrm{NP}$,且任何NP语言A都可以多项式时间归于到B，即$A \leq_P B$.

定理　(Cook-Levin定理)

SAT是NP完备的.

定理

- $HAMPATH$是NP完备的.
- $SUBSET-SUM$是NP完备的,
  其中$SUNSETQ-SUM=\{< X,t >| \ X=\{x_1,x_2,\ldots,x_k\}$ 且有X的子集$\{y_1,y_2,\ldots,y_l\}$使得$\sum_{i=1}^{l} y_i = t\}$.

Back
Close

著名问题

P=NP?

作业

1. 构造TM计算$n-1$函数.
2. 依据下面的图灵机，构造格局演算序列，表明图灵机能接受输入11001♯11001.

   TM能识别语言$B=\{w\sharp w \mid w\in\{0,1\}^*\}$

   设$\Sigma=\{0,1,\sharp\}$, $\Gamma=\{0,1,\sharp,X,\sqcup,\triangleright\}$, 转移函数$\delta$定义如下表：

state	Symbol $\triangleright$	0	1	$\sharp$	X	$\sqcup$
q_0	$(q_1,\triangleright,R)$					
q_1		(q_2,X,R)	(q_3,X,R)	$(q_8,\sharp,R)$		
q_2		$(q_2,0,R)$	$(q_2,1,R)$	$(q_4,\sharp,R)$		
q_3		$(q_3,0,R)$	$(q_3,1,R)$	$(q_5,\sharp,R)$		
q_4		(q_6,X,L)			(q_4,X,R)	
q_5			(q_6,X,L)		(q_5,X,L)	
q_6		$(q_6,0,L)$	$(q_6,1,L)$	(q_7,X,L)	(q_6,X,L)	
q_7		$(q_7,0,L)$	$(q_7,1,l)$		(q_1,X,R)	
q_8					(q_8,X,R)	$(q_a,\sqcup,R)$

3. 修改图灵机TM $M_0 = (\{q_0, q_1, q_2, q_3, q_4 q_a, q_r\}, \{0, 1\}, \{0, 1, X, Y, \triangleright, \sqcup\}, \delta, q_0, q_{accept}, q_{rejct})$

state	Symbol $\triangleright$	0	1	X	Y	$\sqcup$
q_0	$(q_1, \triangleright, R)$					
q_1		(q_2, X, R)	-	-	(q_4, Y, R)	-
q_2		$(q_2, 0, R)$	(q_3, Y, L)	-	(q_2, Y, R)	-
q_3		$(q_3, 0, L)$	-	(q_1, X, R)	(q_3, Y, L)	-
q_4		-	-	-	(q_4, Y, R)	$(q_a, \sqcup, R)$

使之能判定语言: $\{0^n 1^n \mid n \geq 1\}$.