



模式识别

中国科学技术大学 汪增福

- 第一章 绪论
- 第二章 统计模式识别中的几何方法
- 第三章 统计模式识别中的概率方法
- 第四章 分类器的错误率
- 第五章 统计模式识别中的聚类方法
- 第六章 结构模式识别中的句法方法
- 第七章 总结

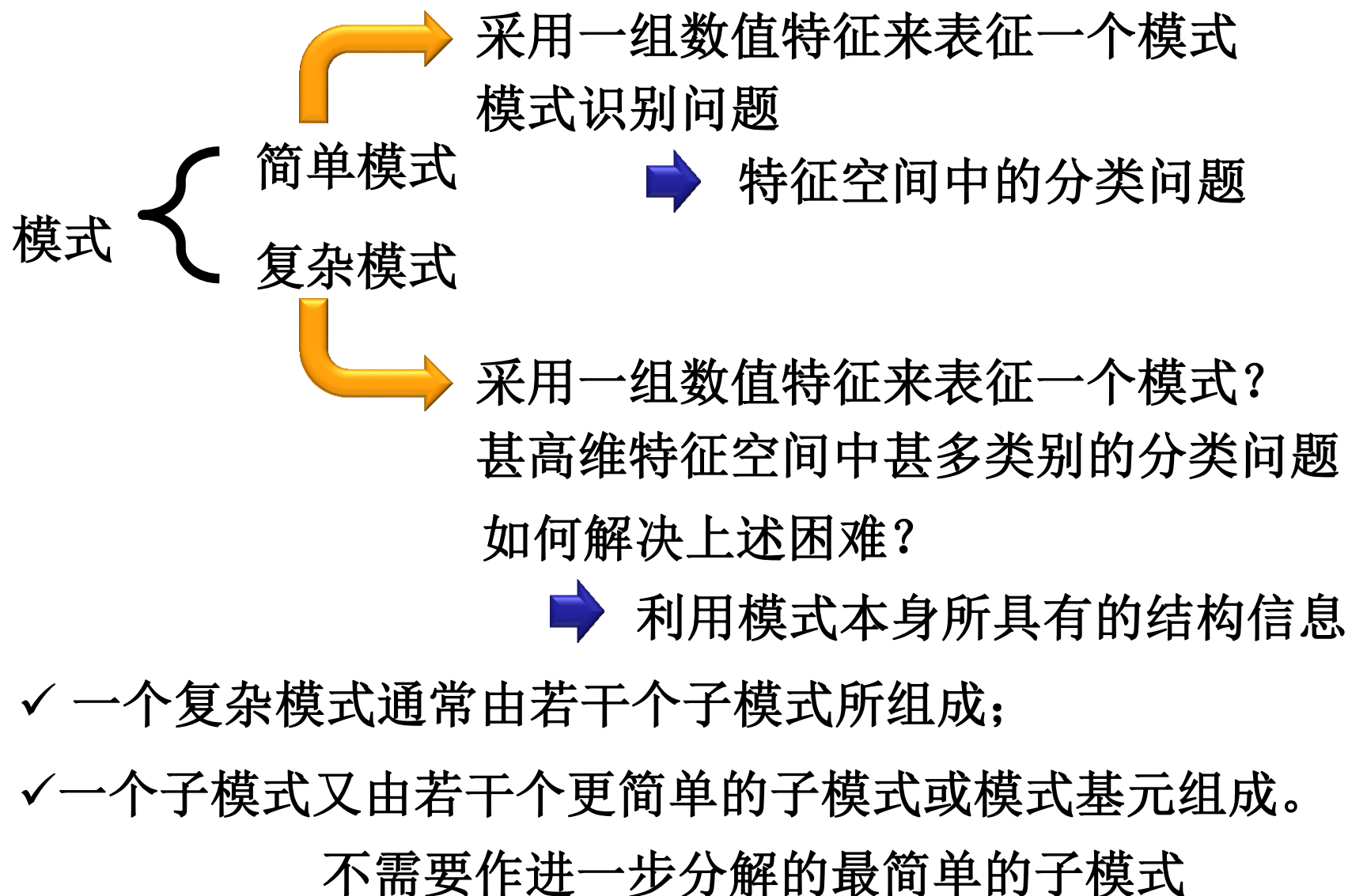
第六章 结构模式识别中的句法方法

● 本章主要内容

主要讨论具有一定结构的复杂模式的识别问题。

- 模式基元和模式结构的表达
- 形式语言基础
- 有限状态自动机
- 下推自动机
- 图灵机
- 句法分析
- 文法推断

§ 6.1 模式基元和模式结构的表达

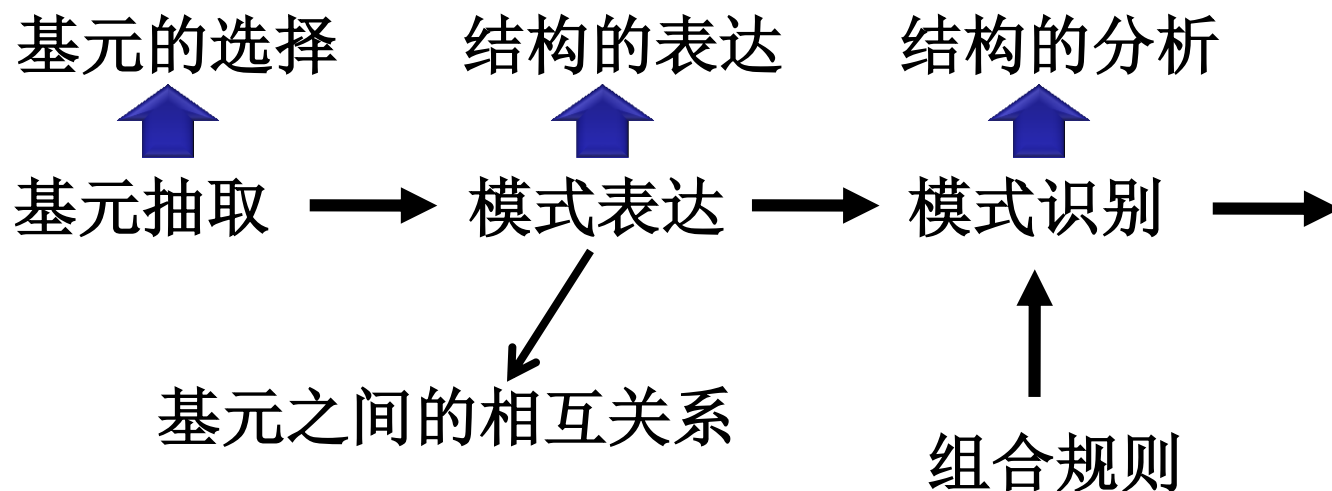


§ 6.1 模式基元和模式结构的表达

- ✓ 一个复杂模式通常由若干个子模式所组成；
- ✓ 一个子模式又由若干个更简单的子模式或模式基元组成。

➡ 复杂模式：由**模式基元**按照一定的**规则组合**而成

➡ 模式具有的结构信息由相应的规则序列所表征

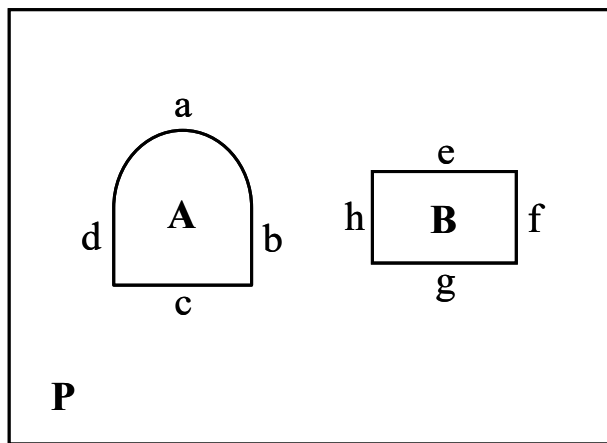


§ 6.1 模式基元和模式结构的表达

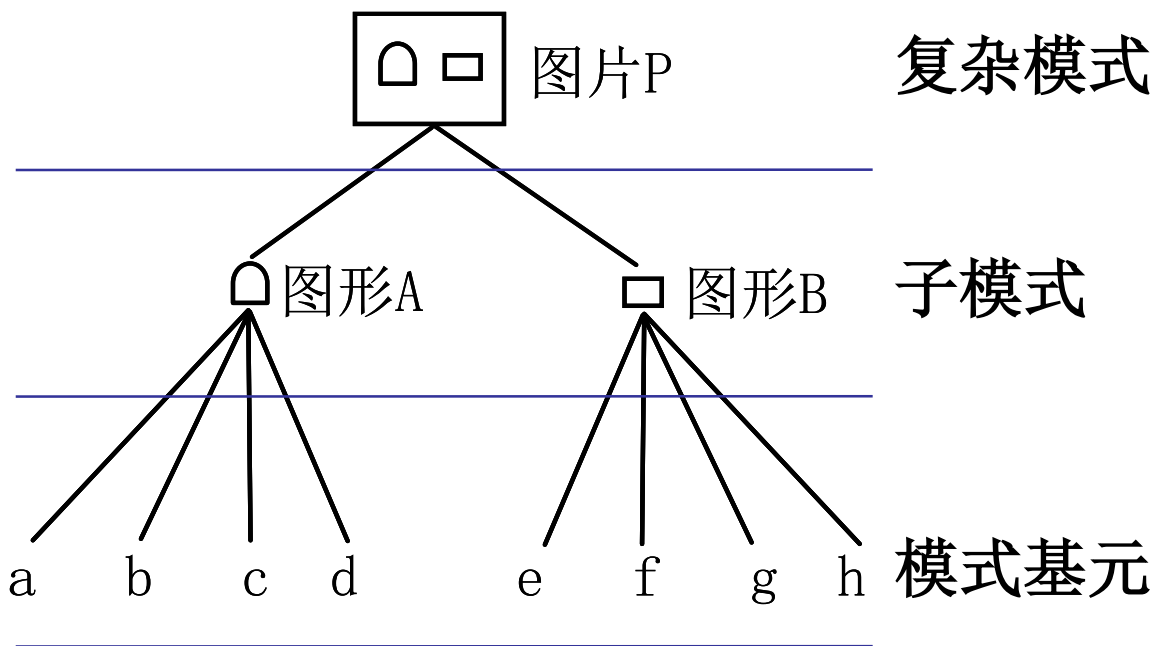
模式基元选择的一般原则

- ✓ 满足特殊应用场合的需求
- ✓ 模式基元本身易于得到

模式基元选择实例

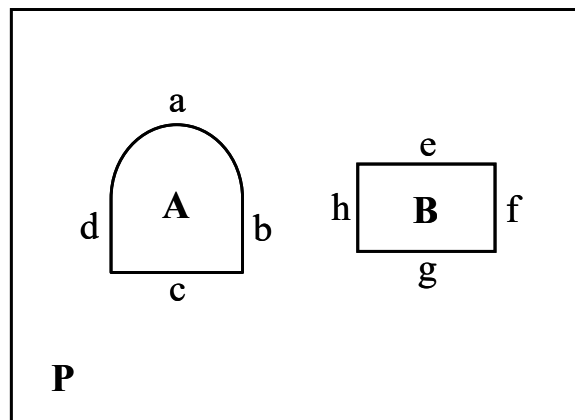


分层结构表示 (一)

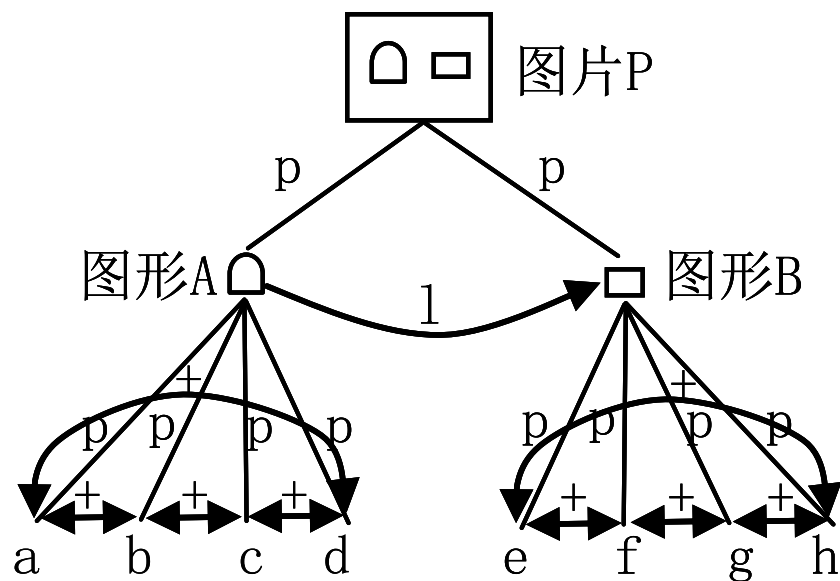


§ 6.1 模式基元和模式结构的表达

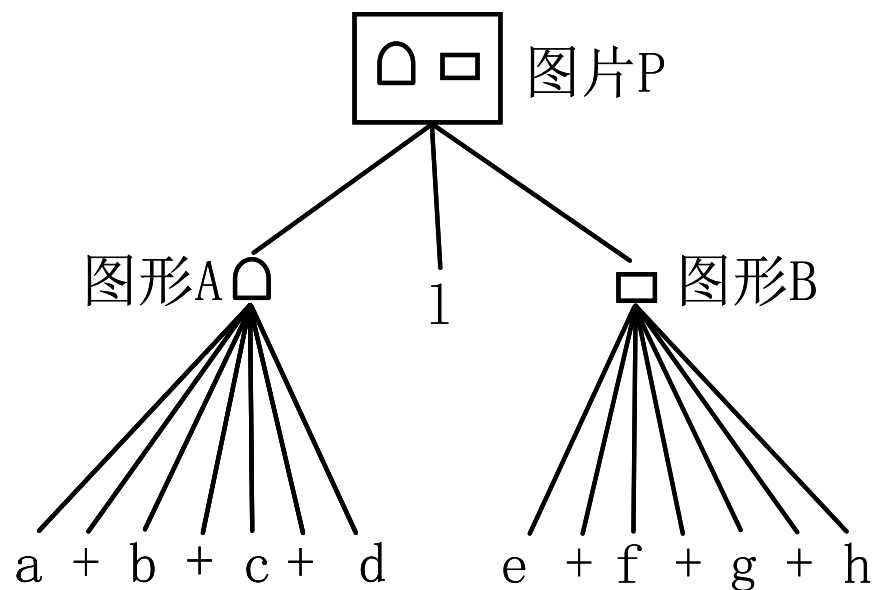
模式基元选择实例



分层结构表示 (二)

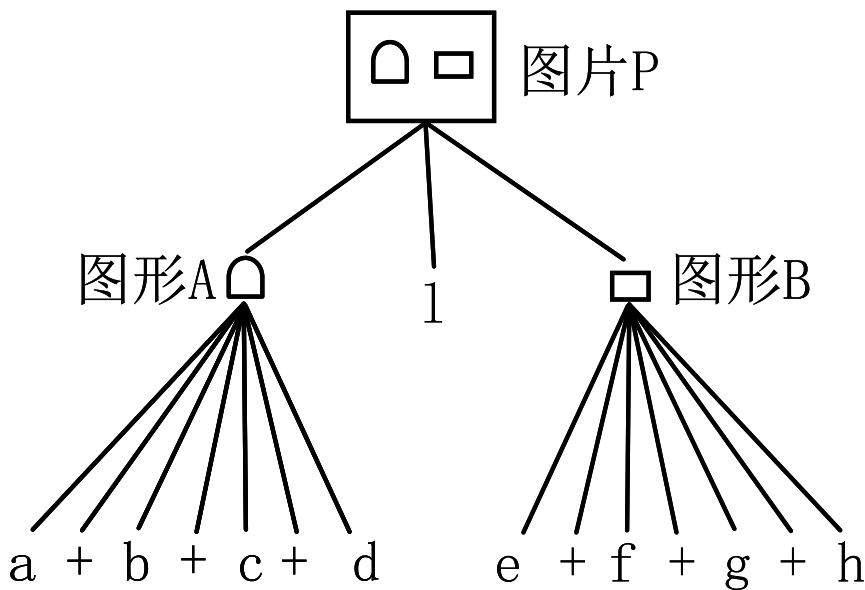
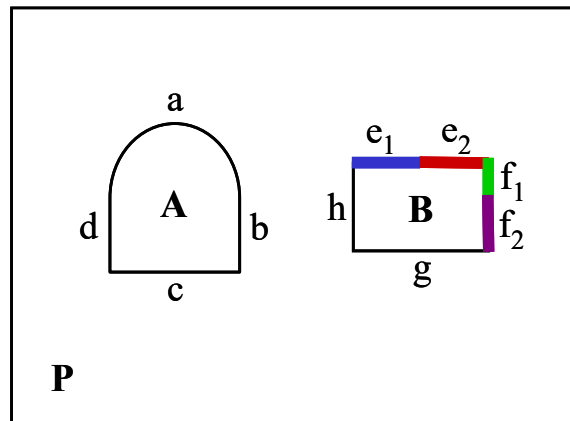
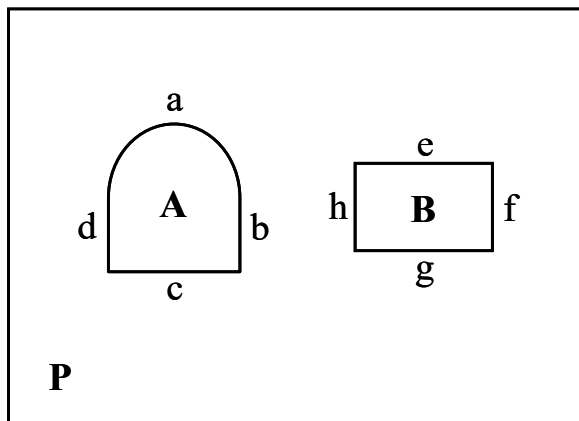


分层结构表示 (三)

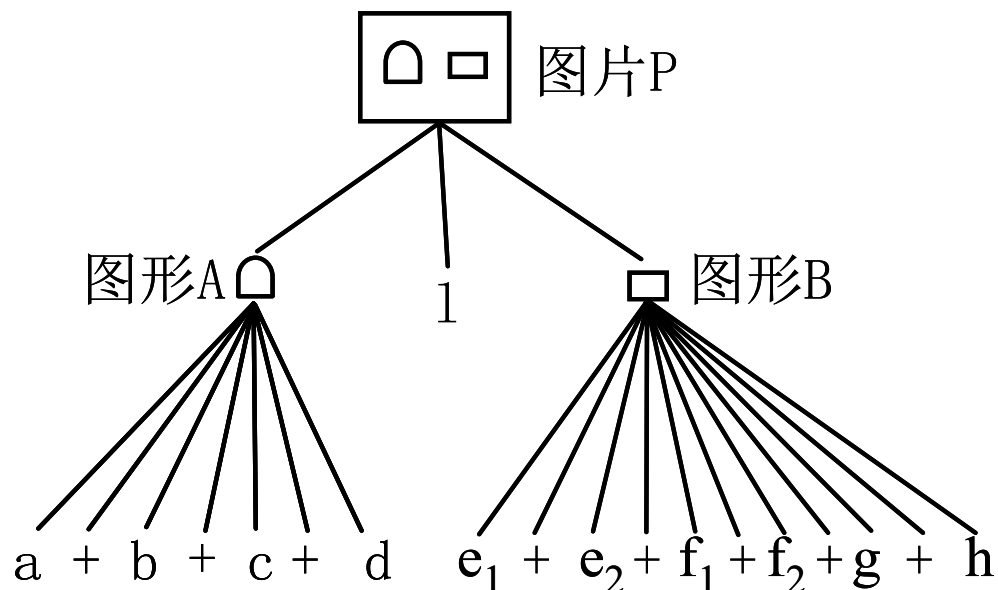


§ 6.1 模式基元和模式结构的表达

模式基元选择实例



理想的分层结构表示



实际的分层结构表示

§ 6.1 模式基元和模式结构的表达

若干结论：

如何对一个模式可能有的多样化的表达进行概括和总结，进而以一种紧凑的方式，通过执行一组操作或适用一组规则形成模式的多样化描述就成为对复杂模式进行识别的重要环节。

如何解决模式的多样化描述问题？

➡ 借鉴文法和语言之间存在的关联性和可类比性

语言由句子所构成，而句子又由单词根据文法所生成。

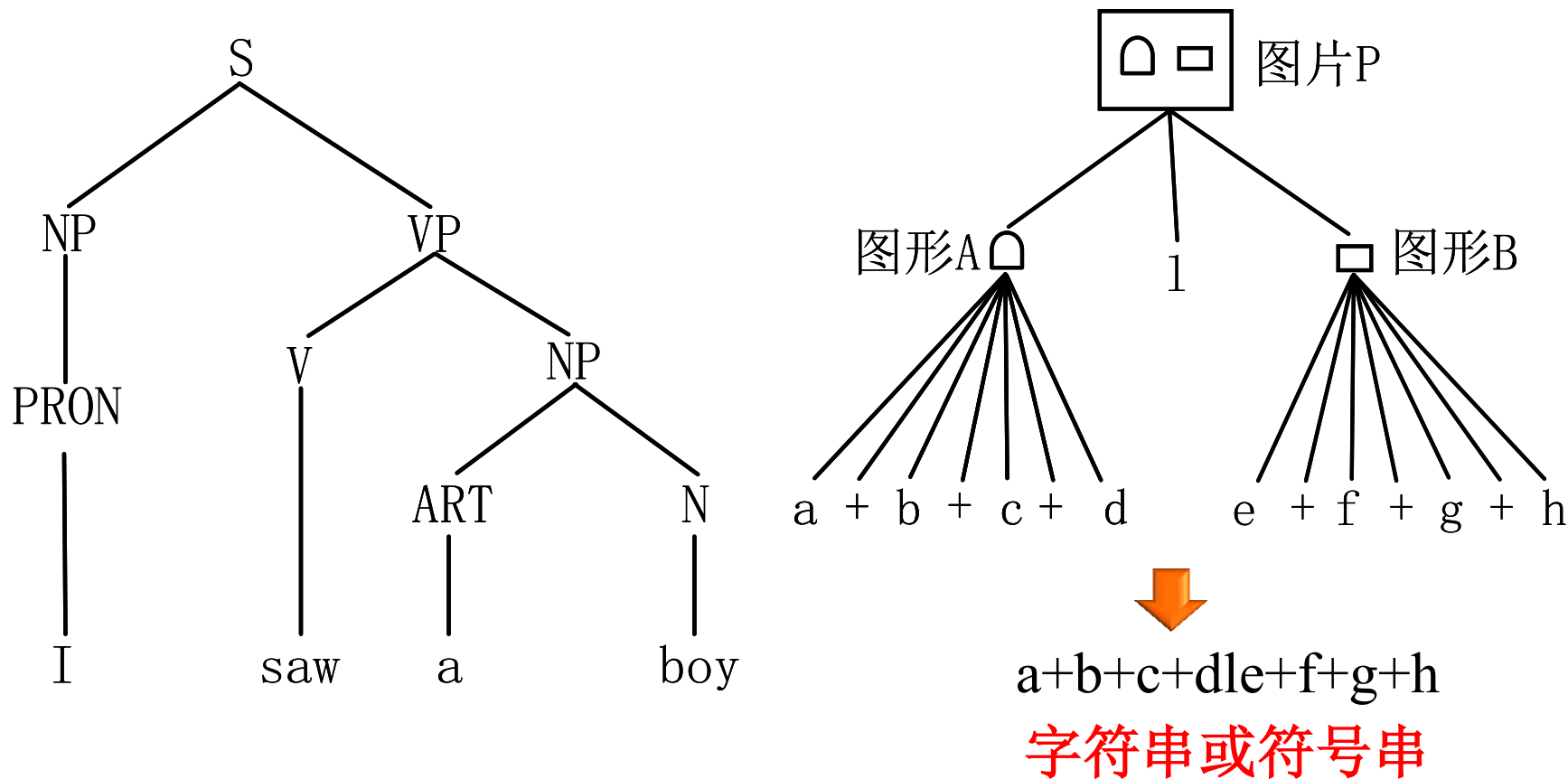
模式类由模式所构成，而模式又由模式基元根据一组装配规则所生成。

➡ 借鉴语言学中业已存在的方法来解决模式的多样化描述和识别问题。

§ 6.1 模式基元和模式结构的表达

句子和模式之间存在的类比关系

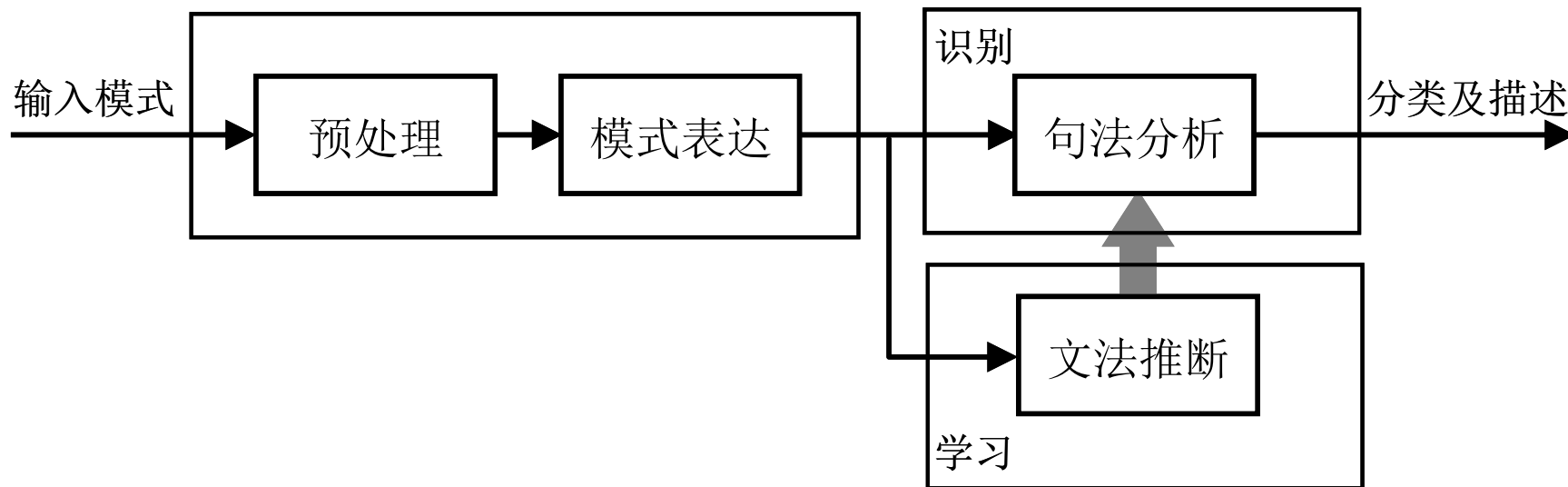
例: **I saw a boy.**



➡ 图片模式和句子属于完全不同的模式范畴，但两者之间存在明显的相似性。

§ 6.1 模式基元和模式结构的表达

句法模式识别系统



1. 学习过程（文法推断）

利用已知结构的样本模式来推断产生这些模式的文法规则。

2. 识别过程（句法分析）

用有序字符串表达输入模式，并利用文法规则对其进行句法分析以判断能否由相应的文法所生成。

§ 6.2 形式语言基础

动机：深入探讨语言的数学结构，以便由此及彼、由表及里。

若干概念

集合：一定范围内，由确定的、彼此之间可以相互区分的对象所形成的整体。

元素：构成集合的成员。

集合的描述：

列举法 $\{a, b, c, d, x, y, z\}$

命题法 $\{x \mid P(x)\}$

集合的分类：

{	有穷集	{	可数无穷集
	无穷集		不可数无穷集

集合的基数：有穷集的基数为其所包含的元素的个数。 $|A|$

空集：基数为0的集合。 Φ

§ 6.2 形式语言基础

若干概念

子集： 设 A, B 是集合, 若 A 的元素均在 B 中, 则称 A 是 B 的子集, 记作 $A \subseteq B$ 。

若 $A \subseteq B$, 且 $\exists x \in B$, 但 $x \notin A$, 则称 A 是 B 的真子集, 记作 $A \subset B$ 。

集合的相等： 若 $A \subseteq B$ 且 $B \subseteq A$, 则 $A = B$ 。

集合的运算：

并 $A \cup B = \{x \mid x \in A \text{ 或 } x \in B\}$

交 $A \cap B = \{x \mid x \in A \text{ 且 } x \in B\}$

差 $A - B = \{x \mid x \in A \text{ 且 } x \notin B\}$

对称差 $A \oplus B = \{x \mid x \in A \text{ 且 } x \notin B \text{ 或者 } x \notin A \text{ 且 } x \in B\}$

笛卡尔乘积 $A \times B = \{(a, b) \mid a \in A \text{ 且 } b \in B\}$

幂集 $2^A = \{B \mid B \subseteq A\}$

§ 6.2 形式语言基础

若干概念

二元关系:

设 A, B 是两个集合, 则称 $R \subseteq A \times B$ 为 A 到 B 的二元关系。其中, A 为定义域, B 为值域。特别地, 当 $A=B$ 时, 称 R 是 A 上的二元关系。

若 $(a,b) \in R$, 则表示 a 与 b 满足关系 R , 记为 aRb 。

若对于 A 中的每一个元素 a , 均在 B 中有唯一的元素 b 使 aRb 成立, 则称 R 为从 A 到 B 的一个映射, 记为 $A \rightarrow B$ 。

关系的三歧性:

设 R 是 A 上的二元关系, 有:

1. 若对 $\forall a \in A$, 都有 $(a, a) \in R$, 则称 R 是自反的。
2. 若对 $\forall a, b \in A$, $(a, b) \in R \Rightarrow (b, a) \in R$, 则称 R 是对称的。
3. 若对 $\forall a, b, c \in A$, $(a, b) \in R$ 且 $(b, c) \in R \Rightarrow (a, c) \in R$, 则称 R 为传递的。

§ 6.2 形式语言基础

若干概念

等价关系：

同时具有自反、对称和传递性质的二元关系称为等价关系。

复合关系：

设 $R \subseteq A \times B$ 是 A 到 B 的二元关系, $S \subseteq B \times C$ 是 B 到 C 的二元关系, 则 A 到 C 的二元关系

$$R \circ S = \{(a, c) \mid a \in A, c \in C \text{ 且 } \exists b \in B, \text{ 使 } (a, b) \in R, (b, c) \in S\}$$

称为 R 与 S 的复合关系。

传递闭包（正闭包）：

设 R 是 A 上的二元关系, 则下述 R^+ 称为 R 的传递闭包。

1. 对 $\forall a, b \in A$, 若 $(a, b) \in R$, 则 $(a, b) \in R^+$ 。
2. 对 $\forall a, b, c \in A$, 若 $(a, b) \in R$ 以及 $(b, c) \in R$, 则 $(a, c) \in R^+$ 。

自反传递闭包（克林闭包）：

设 R 是 A 上的二元关系, 则下述 R^* 称为 R 的自反传递闭包。

$$R^* = R^+ \cup \{(a, a) \mid a \in A\}$$

§ 6.2 形式语言基础

若干概念

关系的n次幂:

设 R 是 A 上的二元关系, 则 R^n 可如下递归定义:

1. $R^0 = \{(a, a) \mid a \in A\}$

2. $R^1 = R$

3. $R^n = R^{n-1} \circ R \ (n = 2, 3, \dots)$

➡ $R^+ = R \cup R^2 \cup R^3 \dots$
 $R^* = R^0 \cup R \cup R^2 \cup R^3 \dots$

➡ 当 A 为有穷集时, 有:

$$R^+ = R \cup R^2 \cup R^3 \dots \cup R^{|A|}$$
$$R^* = R^0 \cup R \cup R^2 \cup R^3 \dots \cup R^{|A|}$$

§ 6.2 形式语言基础

若干概念

字母表：一个非空的有穷集合，用 Σ 进行标记。

整体性-不关心是否还有更细致的结构

可辨认性-可用非语言学的方法得到

字母表的乘积：

$$\Sigma_1 \Sigma_2 = \{ ab \mid a \in \Sigma_1 \text{ 且 } b \in \Sigma_2 \}$$

字母表的幂：

$$\Sigma^n = \Sigma^{n-1} \Sigma, \text{ 当 } n \geq 1 \text{ 时}$$

字母表的传递闭包：

$$\Sigma^+ = \Sigma \cup \Sigma^2 \cup \Sigma^3 \dots$$

字母表的自反传递闭包：

$$\Sigma^* = \Sigma^0 \cup \Sigma \cup \Sigma^2 \cup \Sigma^3 \dots$$

§ 6.2 形式语言基础

若干概念

符号串（链、句子）：由 Σ 中的符号组成的任意有穷序列。

符号串的长度：符号串中所包含的字符的个数。 $|x|$

符号串的链接：将两个符号串首尾链接形成的新符号串。

$$xy |xy| = |x| + |y|$$

链接运算的性质

$$1. (xy)z = x(yz)$$

$$2. xy = xz \Rightarrow y = z$$

$$3. yx = zx \Rightarrow y = z$$

空串：不包含任何符号的串，记为 λ 。 $\lambda x = x\lambda = x$

零串：链接运算中的零元素，记为 ϕ 。 $\phi x = x\phi = \phi$

子串与前、后缀： $w = xyz$

§ 6.2 形式语言基础

若干概念

符号串的幂: $x^0 = \lambda$
 $x^n = x^{n-1}x$, 当 $n \geq 1$ 时

符号串集合的链接: $AB = \{x_i y_j \mid x_i \in A \text{ 且 } y_j \in B\}$

语言: 定义在字母表 Σ 上的句子的集合。 $L \subseteq \Sigma^*$

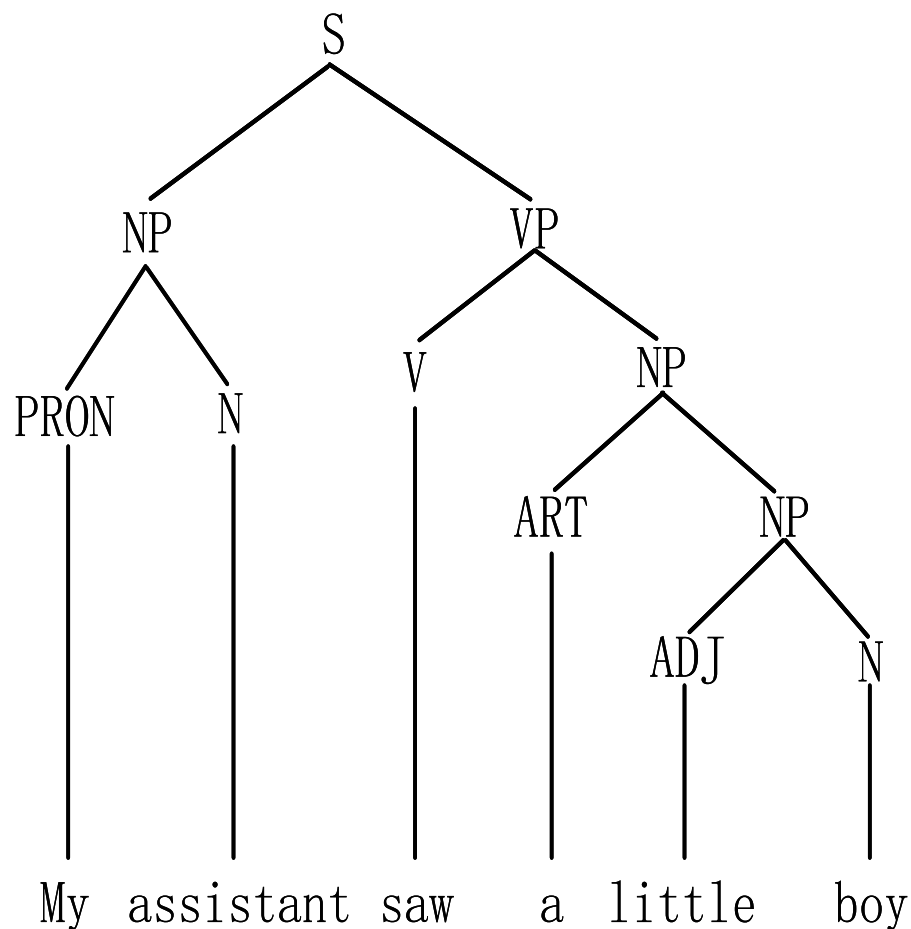
语言的链接: $L_1 L_2 = \{xy \mid x \in L_1, y \in L_2\}$

语言的闭包: $L^+ = L \cup L^2 \cup L^3 \dots$
 $L^* = L^0 \cup L \cup L^2 \cup L^3 \dots$

§ 6.2 形式语言基础

文法

My assistant saw a little boy



<句子> 根节点**S**

<名词短语> 中间节点**NP**

<动词短语> 中间节点**VP**

<代词> 中间节点**PRON**

<名词> 中间节点**N**

<动词> 中间节点**V**

<冠词> 中间节点**ART**

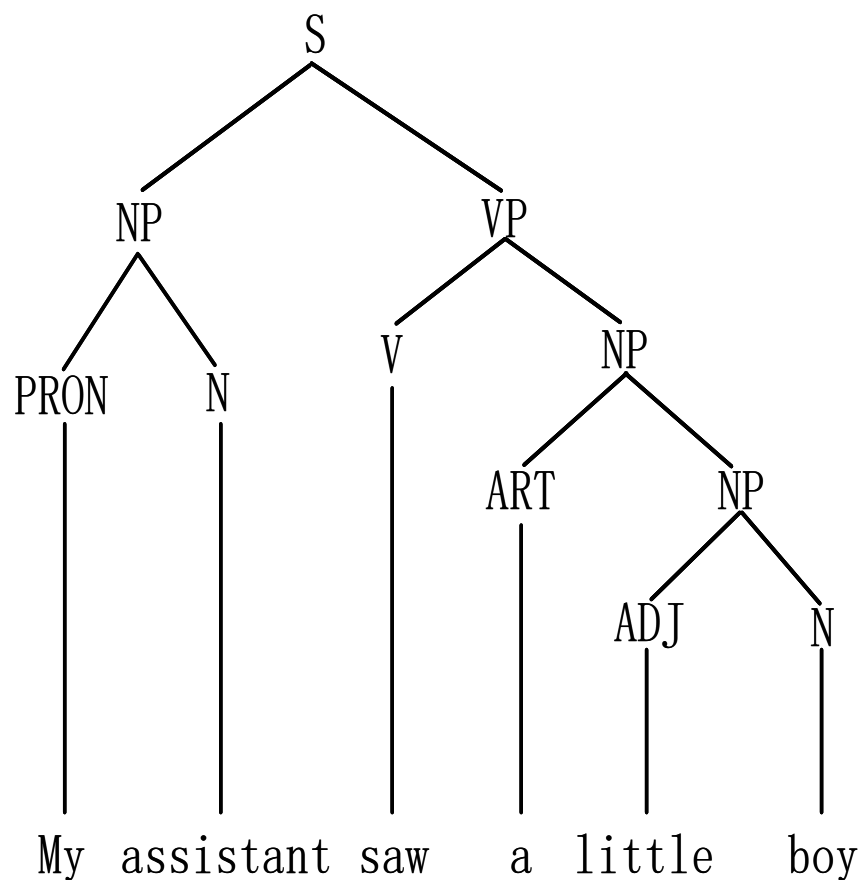
<形容词> 中间节点**ADJ**

.....

§ 6.2 形式语言基础

文法

My assistant saw a little boy



再写规则

起始符
可改写为
非终结符

<句子> → <名词短语><动词短语>

<名词短语> → <代词><名词>

<动词短语> → <动词><名词短语>

<名词短语> → <冠词><名词短语>

<名词短语> → <形容词><名词>

<代词> → my

<名词> → assistant

<动词> → saw

<冠词> → a

<形容词> → little

<名词> → boy

终结符

§ 6.2 形式语言基础

文法

定义 文法 $\mathbf{G} = (\mathbf{N}, \mathbf{T}, \mathbf{P}, \mathbf{S})$ 是一个四元式。其中, \mathbf{N} 为 \mathbf{G} 的非终结符或变量的有穷集合, \mathbf{T} 为 \mathbf{G} 的终结符或常量的有穷集合, \mathbf{P} 是产生式或再写规则的有穷集合, 而 $\mathbf{S} \in \mathbf{N}$ 为句子的起始符。

$$N \cap T = \Phi \quad \Sigma = N \cup T \quad \alpha \rightarrow \beta$$

一些约定

大写的拉丁字母 非终结符

小写的拉丁字母 终结符

小写的希腊字母 由非终结符和终结符组成的串

导出=推导=派生

\Rightarrow_G $\Sigma = N \cup T$ 上的一个二元关系

$$\begin{array}{ccc} + & * & n \\ \Rightarrow_G & \Rightarrow_G & \Rightarrow_G \end{array}$$

§ 6.2 形式语言基础

句型、语言与句子

定义 设有文法 $\mathbf{G} = (\mathbf{N}, \mathbf{T}, \mathbf{P}, \mathbf{S})$, 对 $\forall \alpha \in \Sigma^*$, 如果有 $S \xRightarrow[G]{*} \alpha$, 则称 α 是由产生的一个句型。

定义 设有文法 $\mathbf{G} = (\mathbf{N}, \mathbf{T}, \mathbf{P}, \mathbf{S})$, 称_{*}

$$L(G) = \{ x \mid x \in T^* \text{ 且 } S \Rightarrow x \}$$

为由文法所产生的语言。

$x \in L(G)$ 为由文法产生的句子

§ 6.2 形式语言基础

句型、语言与句子

举例

$$G = (N, T, P, S)$$

$$N = \{S\}$$

$$T = \{a, b\}$$

$$P: (1) S \rightarrow aS$$

$$(2) S \rightarrow b$$

导出过程

$$\begin{array}{c} (2) \\ S \Rightarrow_G b \end{array}$$

$$\begin{array}{cc} (1) & (2) \\ S \Rightarrow_G aS \Rightarrow_G ab \end{array}$$

$$\begin{array}{ccc} (1) & (1) & (2) \\ S \Rightarrow_G aS \Rightarrow_G aaS \Rightarrow_G aab \end{array}$$

$$\begin{array}{cccc} (1) & (1) & (1) & (2) \\ S \Rightarrow_G aS \Rightarrow_G aaS \Rightarrow_G aaaS \Rightarrow_G aaab \end{array}$$

$$L(G) = \{b, ab, aab, aaab, \dots\} = \{x \mid x = a^n b, n \geq 0\}$$

§ 6.2 形式语言基础

句型、语言与句子 举例

导出过程

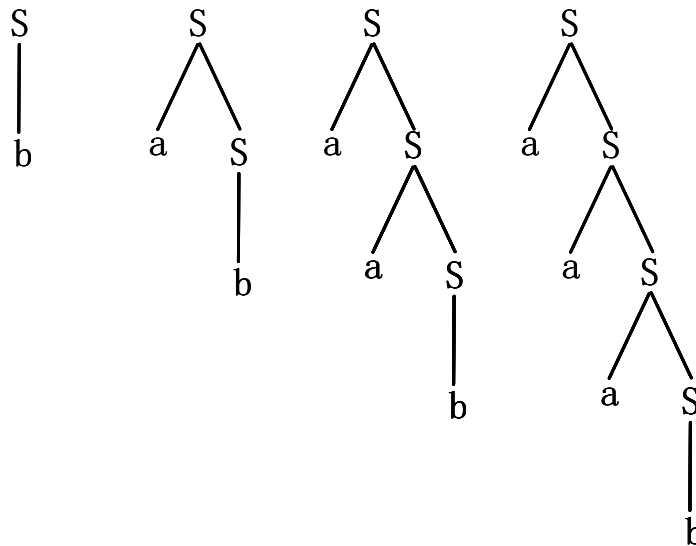
$$\begin{array}{c} (2) \\ S \Rightarrow b \\ \quad G \end{array}$$

$$\begin{array}{cc} (1) & (2) \\ S \Rightarrow aS \Rightarrow ab \\ \quad G & \quad G \end{array}$$

$$\begin{array}{ccc} (1) & (1) & (2) \\ S \Rightarrow aS \Rightarrow aaS \Rightarrow aab \\ \quad G & \quad G & \quad G \end{array}$$

$$\begin{array}{cccc} (1) & (1) & (1) & (2) \\ S \Rightarrow aS \Rightarrow aaS \Rightarrow aaaS \Rightarrow aaab \\ \quad G & \quad G & \quad G & \quad G \end{array}$$

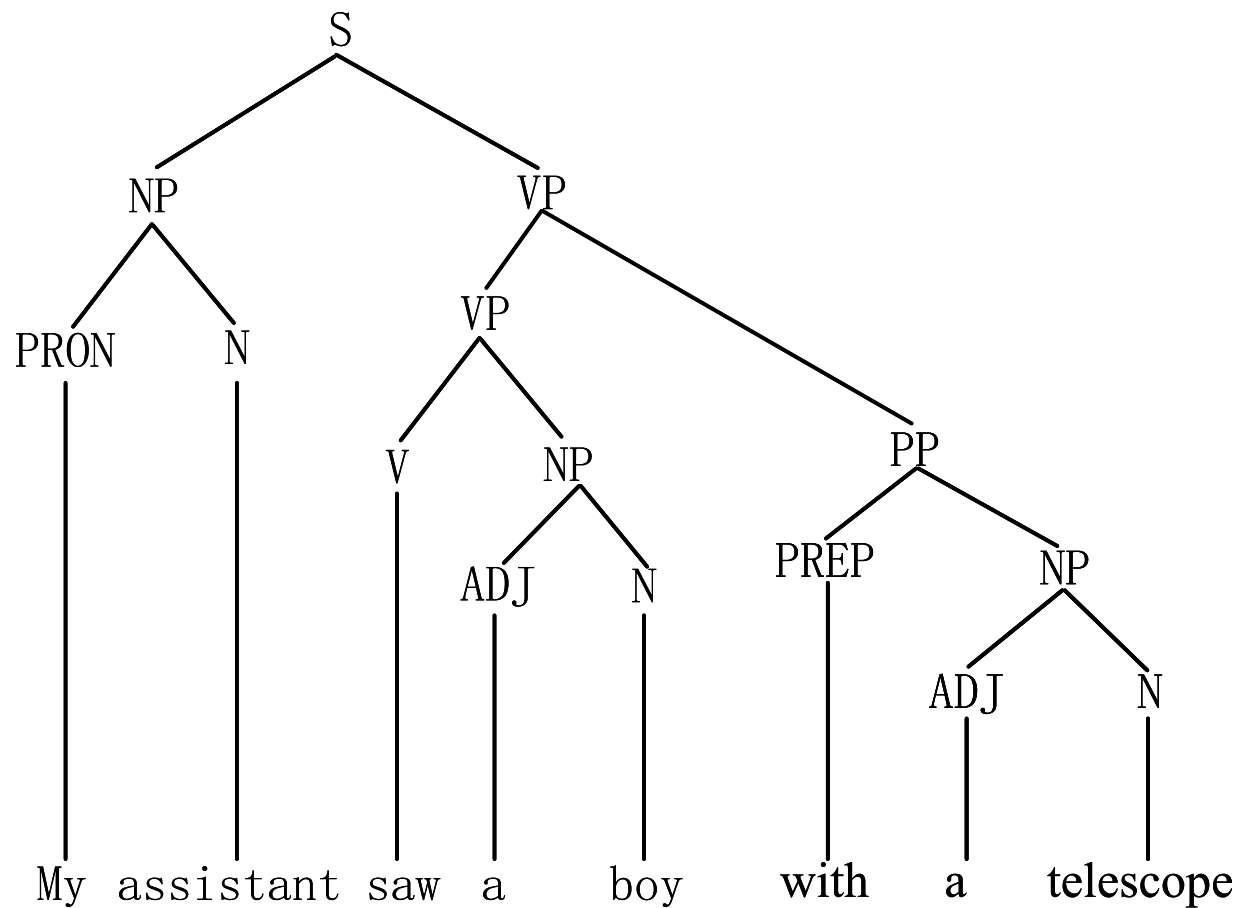
导出树



§ 6.2 形式语言基础

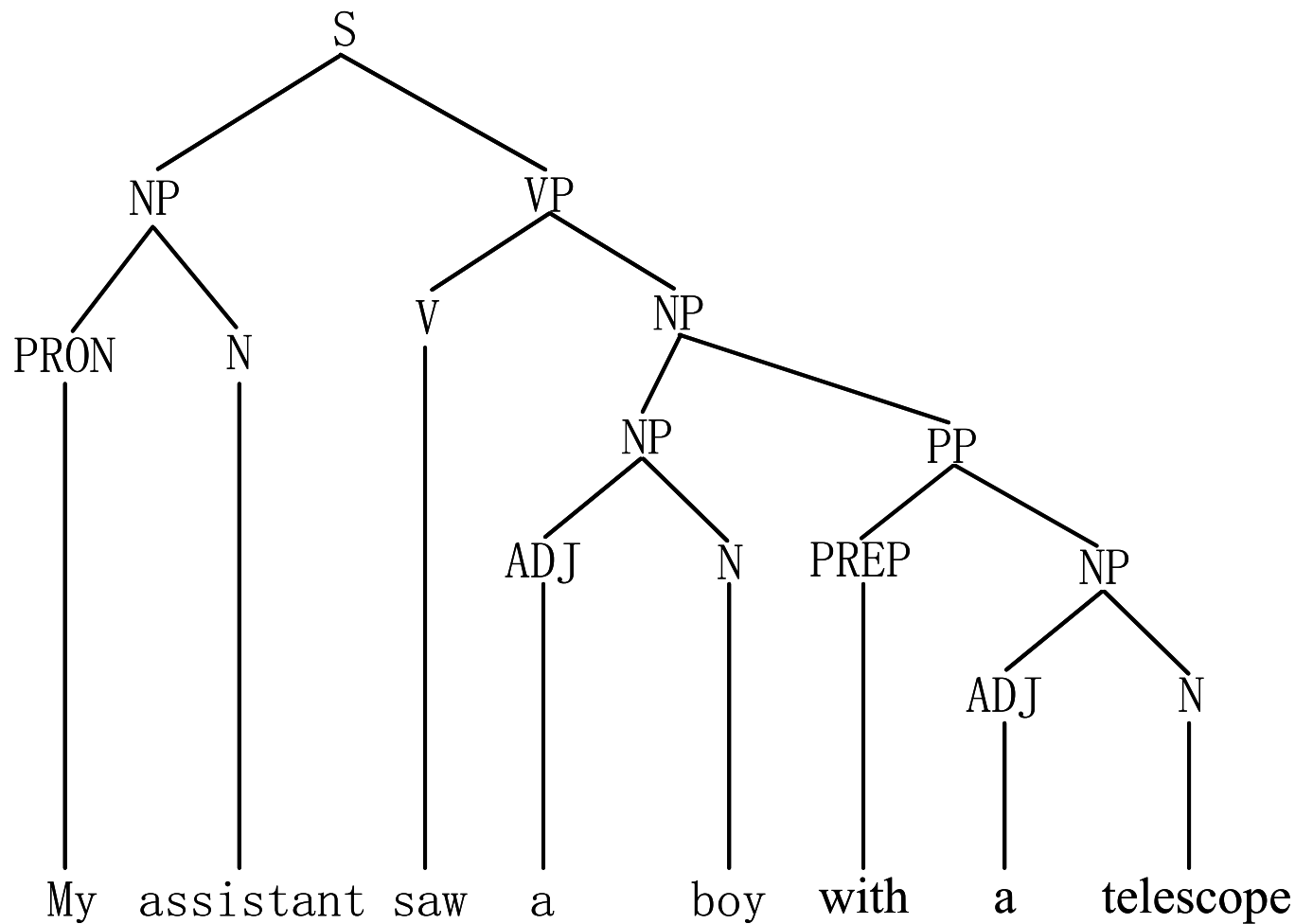
文法的二义性

My assistant saw a boy with a telescope.



§ 6.2 形式语言基础

文法的二义性



§ 6.2 形式语言基础

文法的二义性举例

冬天：能穿多少穿多少；夏天：能穿多少穿多少。

剩女产生的原因有两个，一是谁都看不上，二是谁都看不上。

公交车里听到一个女孩大概是给男朋友打电话。

“我已经到科大西区北大门了，你赶紧出来往北大门走吧。”

“如果你到了，我还没到，你就等着吧。

如果我到了，你还没到，你就等着吧。”

单身人的来由：原来是喜欢一个人，现在是喜欢一个人。

一中年男子看到雕像【岳母刺字】，感叹道：也就是岳母干的出来，亲妈不能干这事。

§ 6.2 形式语言基础

文法的二义性举例

老外苦学汉语十年，到中国参加[汉语等级考试](#)，试题如下：

请解释下文中每个“意思”的意思——

阿呆给领导送红包时，两人的对话颇有意思。

领导：“你这是什么意思？”

阿呆：“没什么意思，意思意思。”

领导：“你这就不够意思了。”

阿呆：“小意思，小意思。”

领导：“你这人真有意思。”

阿呆：“其实也没有别的意思。”

领导：“那我就不好意思了。”

阿呆：“[是我不好意思](#)。”

老外泪流满面，交白卷回国了。

§ 6.2 形式语言基础

文法的二义性举例

天热，公交大巴换空调车了，票价由原来的一块调为两块。

大妈上车投了一块。 司机：两块啊。

大妈点头：嗯，凉快。

司机说：投两块！

大妈笑曰：不光头凉快，浑身都凉快。

说完大妈径直往后头走…

司机急了大喊：钱投两块！

大妈：后头人少更凉快……

§ 6.2 形式语言基础

文法的二义性举例

早上出门，老公发现车钥匙下面压了一张老婆写的小纸条。

上写：老公，加油！ ❤️

当时老公就泪奔了！老夫老妻的啦，还这么励志。

心里默默发誓：老婆，我一定会加倍努力的。



车开了15分钟后，车停下来了。

为啥？

因为车没油了。

§ 6.2 形式语言基础

文法的分类

0型文法 无约束文法/短语结构文法

产生式具有 $\alpha \rightarrow \beta$ 形式的文法称为**0型文法**。

$$\alpha \in \Sigma^+ \quad \beta \in \Sigma^*$$

1型文法 上下文有关文法

产生式具有 $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$ 形式的文法称为**1型文法**。

$$\alpha_1, \alpha_2 \in \Sigma^* \quad A \in N \quad \beta \in \Sigma^+ \quad |A| < |\beta|$$

2型文法 上下文无关文法

产生式具有 $A \rightarrow \beta$ 形式的文法称为**2型文法**。

$$A \in N \quad \beta \in \Sigma^+$$

3型文法 有限状态文法/正则文法

产生式具有 $A \rightarrow aB$ 或 $A \rightarrow a$ 形式的文法称为**3型文法**。

$$A, B \in N \quad a \in T$$

§ 6.2 形式语言基础

文法之间的关系

从表现力看

0型文法 > **1型文法** > **2型文法** > **3型文法**

从规范性看

0型文法 < **1型文法** < **2型文法** < **3型文法**

从包含关系看

0型文法 \supset **1型文法** \supset **2型文法** \supset **3型文法**

文法的命名

依据规范性顺序看给定文法是否满足相关文法的条件

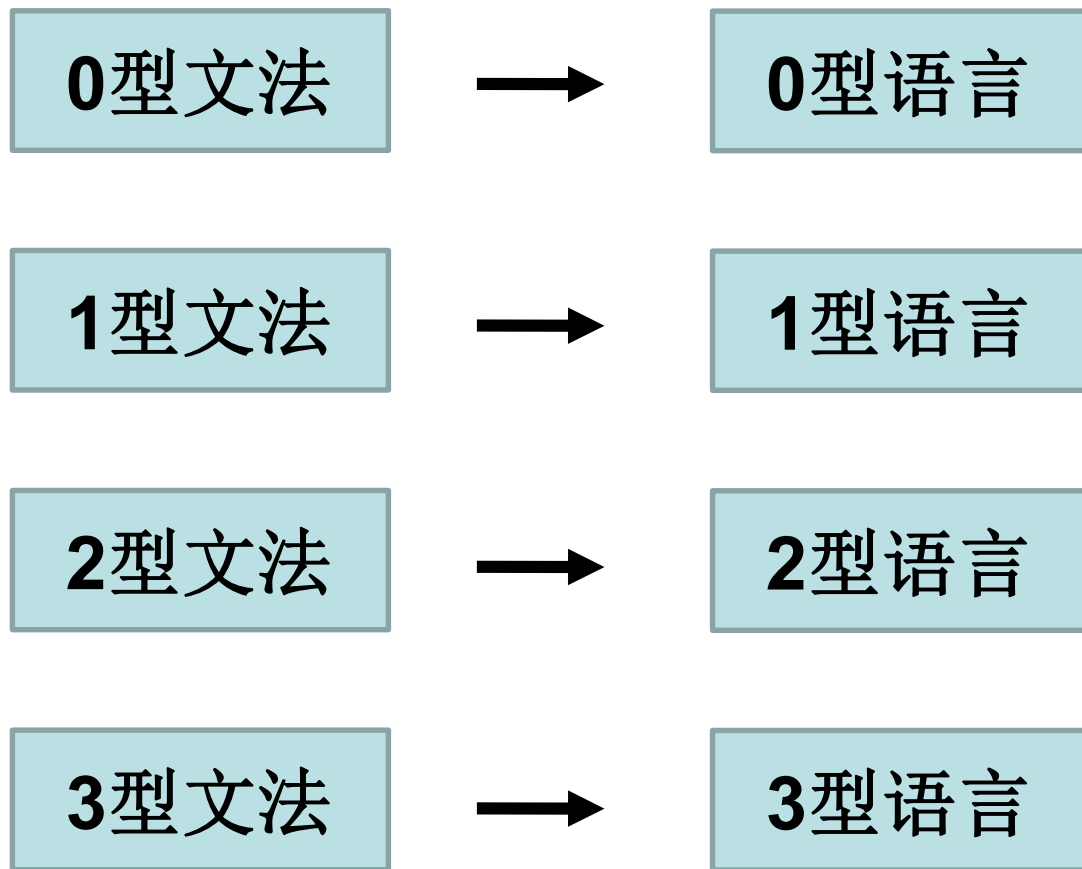
语言的命名

依据产生语言的文法命名

如果一个语言可由若干种文法所产生，则将该语言命名为由最规范的文法所产生的语言。

§ 6.2 形式语言基础

文法的分类



§ 6.2 形式语言基础

上下文无关文法与导出树

举例说明:

$$G = (N, T, P, S)$$

$$N = \{S, A, B\}$$

$$T = \{a, b\}$$

$$P: 1) S \rightarrow aB$$

$$2) S \rightarrow bA$$

$$3) A \rightarrow a$$

$$4) A \rightarrow aS$$

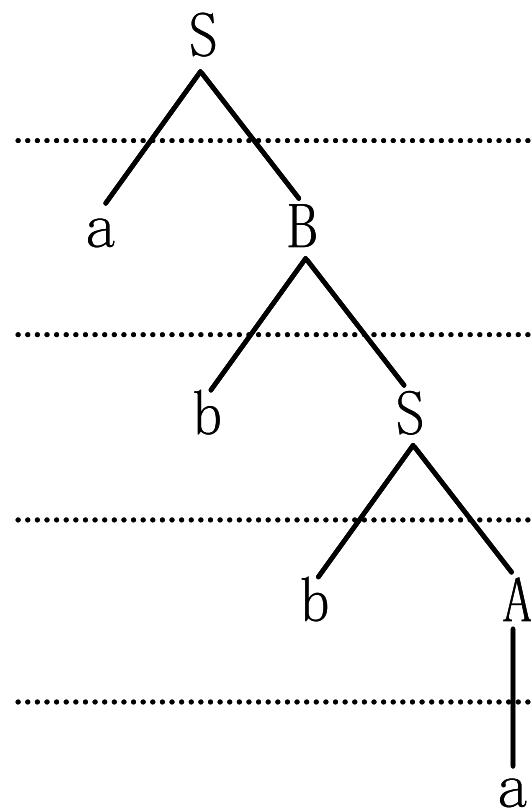
$$5) A \rightarrow bAA$$

$$6) B \rightarrow b$$

$$7) B \rightarrow bS$$

$$8) B \rightarrow aBB$$

串 *abba* 的导出树



被选中的产生式

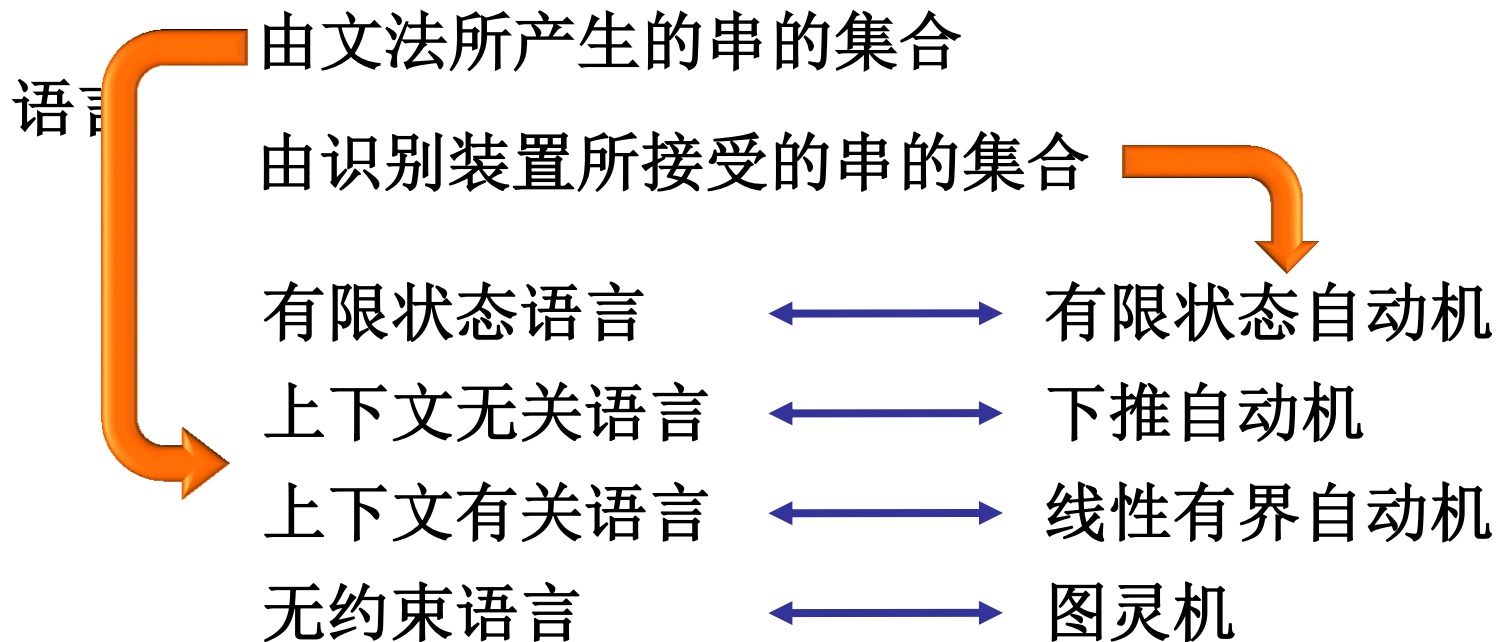
$$(1) S \rightarrow aB$$

$$(7) B \rightarrow bS$$

$$(2) S \rightarrow bA$$

$$(3) A \rightarrow a$$

§ 6.3 有限状态自动机



问题：适当设计的有限状态自动机可识别无约束语言吗？

问题：适当设计的下推自动机可识别有限状态语言吗？

问题：适当设计的图灵机可识别上下文无关语言吗？

问题：适当设计的图灵机可识别有限状态语言吗？

§ 6.3 有限状态自动机

确定的有限状态自动机

一个确定的有限状态自动机 (**DFA**) 是一个五元式:

$$A_f = (Q, \Sigma_I, \delta, q_0, F)$$

Q 状态的有限集合

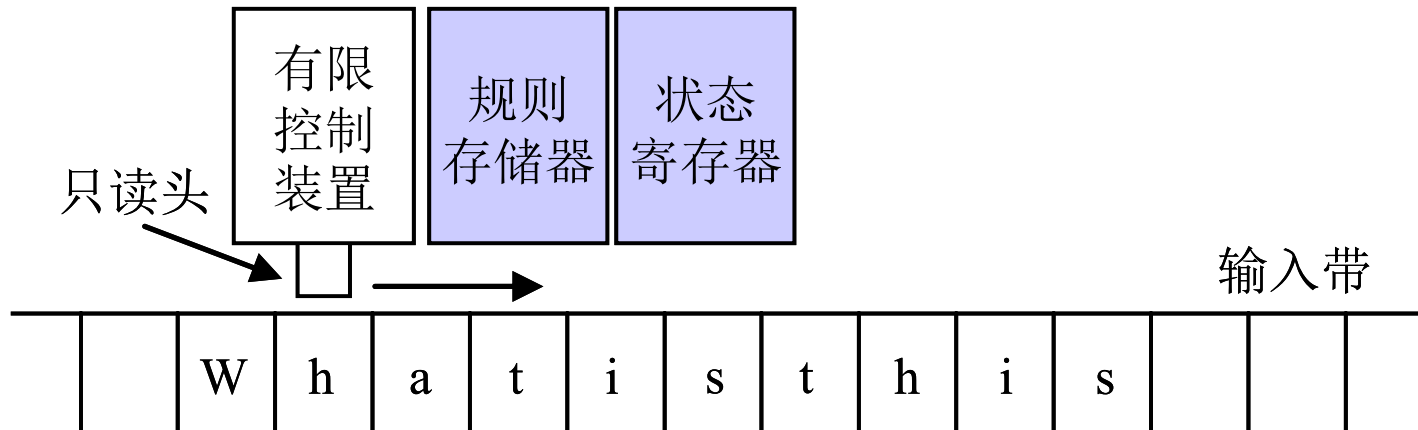
Σ_I 输入符号的有限集合, 即字母表

δ 映射规则 $\delta: Q \times \Sigma_I \rightarrow Q$

$q_0 \in Q$ 初始状态

$F \subseteq Q$ 终止状态集合

五元式表示

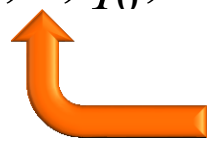


§ 6.3 有限状态自动机

确定的有限状态自动机

$$A_f = (Q, \Sigma_I, \delta, q_0, F)$$

核心


$$\delta : Q \times \Sigma_I \rightarrow Q$$

规定了在当前状态和当前输入符号下
有限状态自动机所执行的操作

$\delta(q, a) = q'$ 状态转移函数

在当前符号为 a 的情况下将有限状态自动机从当前状态 q 映射到下一个状态 q'

有限状态自动机可由一组状态转移函数所完全描述。

对一个确定的有限状态自动机而言，若其状态数为 m ，字母表所含符号个数为 n ，则为了完全描述该自动机的行为，需要定义 $m \times n$ 个状态转移函数。

§ 6.3 有限状态自动机

确定的有限状态自动机的状态转移表表示

举例说明：

$$A_f = (Q, \Sigma_I, \delta, q_0, F)$$

$$Q = \{q_0, q_1, q_2, q_3\} \quad \Sigma_I = \{0, 1\} \quad q_0 \in Q \quad F = \{q_0\}$$

$$\delta(q_0, 0) = q_1 \quad \delta(q_0, 1) = q_2 \quad \delta(q_1, 0) = q_0 \quad \delta(q_1, 1) = q_3$$

$$\delta(q_2, 0) = q_3 \quad \delta(q_2, 1) = q_0 \quad \delta(q_3, 0) = q_2 \quad \delta(q_3, 1) = q_1$$

在状态名的左上角
打一个小点，表示
初始状态

在状态名的右下角
打一个小点，表示
终止状态

符号 状态	0	1
$\cdot q_0$	q_1	q_2
q_1	$\cdot q_0$	q_3
q_2	q_3	$\cdot q_0$
q_3	q_2	q_1

§ 6.3 有限状态自动机

确定的有限状态自动机的有向图表示

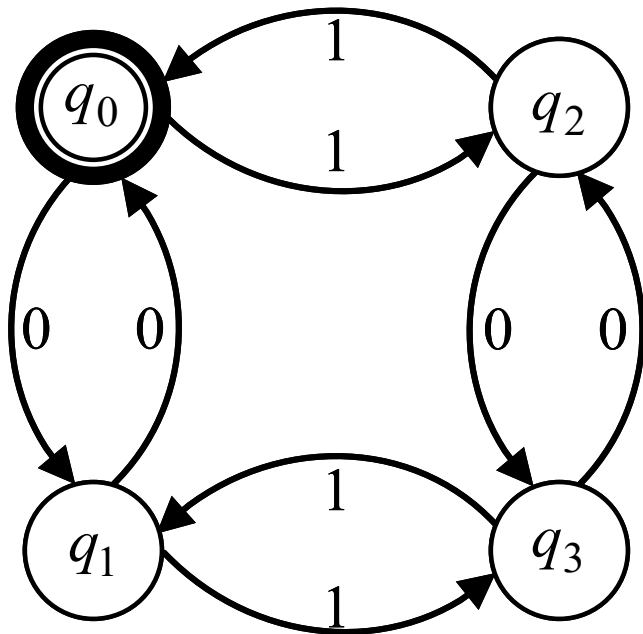
举例说明：

$$A_f = (Q, \Sigma_I, \delta, q_0, F)$$

$$Q = \{q_0, q_1, q_2, q_3\} \quad \Sigma_I = \{0, 1\} \quad q_0 \in Q \quad F = \{q_0\}$$

$$\delta(q_0, 0) = q_1 \quad \delta(q_0, 1) = q_2 \quad \delta(q_1, 0) = q_0 \quad \delta(q_1, 1) = q_3$$

$$\delta(q_2, 0) = q_3 \quad \delta(q_2, 1) = q_0 \quad \delta(q_3, 0) = q_2 \quad \delta(q_3, 1) = q_1$$

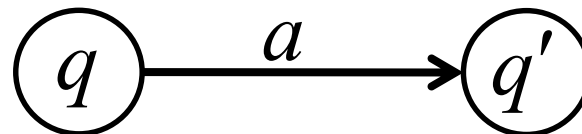


加粗线圈的状态名 初始状态

加双圈的状态名 终结状态

带圈的状态 有向图的节点

$$\delta(q, a) = q'$$



§ 6.3 有限状态自动机

有限状态自动机的任务和工作过程

任务：利用有限控制从左到右顺序地从输入带上读出符号，并根据映射规则确定每一时刻自动机所处的状态，当扫描完输入符号串时根据当前状态决定是否接受输入符号串。

确定的有限状态自动机的工作三部曲

初始操作：在初始时刻，使有限状态自动机处于初始状态，并使其只读头指向带上符号串的最左一个符号。

中间处理：根据当前符号和当前状态，利用 δ 映射规则将自动机映射到下一个状态，并控制其只读头右移一个单元。

判决处理：顺序扫描输入符号串直到下列事态之一出现。

1. 在当前状态和当前符号下，无映射规则可用。
2. 扫描完输入符号串，有限状态自动机停留在一个状态上。
停机，并拒识或判断不可接受。
若停留在终结状态，则判决可接受，否则判断不可接受。

§ 6.3 有限状态自动机

映射规则的推广

$$\delta : Q \times \Sigma_I \rightarrow Q \quad \text{只适用于单个输入符号}$$



$$\delta^* : Q \times \Sigma_I^* \rightarrow Q$$



由单个输入符号的映射规则如下递归定义得到

$$\delta^*(q, \lambda) = q$$

$$\delta^*(q, xa) = \delta^*(\delta^*(q, x), a)$$

$$\delta^*(q, a) = \delta(q, a)$$

$$x \in \Sigma_I^* \quad a \in \Sigma_I \quad q \in Q$$

$$\delta^*(q, x) = q'$$

§ 6.3 有限状态自动机

映射规则的推广

$$x = a_1 a_2 a_3 \dots a_{n-2} a_{n-1} a_n$$

$$\delta^*(q, x) = \delta^*(q, a_1 a_2 a_3 \dots a_{n-2} a_{n-1} a_n)$$

$$= \delta^*(\delta^*(q, a_1 a_2 a_3 \dots a_{n-2} a_{n-1}), a_n)$$

$$= \delta^*(\delta^*(\delta^*(q, a_1 a_2 a_3 \dots a_{n-2}), a_{n-1}), a_n)$$

$$= \delta^*(\delta^*(\delta^* \dots \delta^*(\delta^*(\delta^*(q, a_1), a_2) a_3) \dots a_{n-2}), a_{n-1}), a_n)$$

$$= \delta(\delta(\delta \dots \delta(\delta(\delta(q, a_1), a_2) a_3) \dots a_{n-2}), a_{n-1}), a_n)$$

不妨设

$$\delta(q, a_1) = q_1$$

$$\delta(q_1, a_2) = q_2$$

.....

$$\delta(q_{n-2}, a_{n-1}) = q_{n-1}$$

$$\delta(q_{n-1}, a_n) = q'$$

§ 6.3 有限状态自动机

映射规则的推广

$$\begin{aligned}\delta^*(q, x) &= \delta^*(q, a_1 a_2 a_3 \dots a_{n-2} a_{n-1} a_n) \\ &= \delta^*(\delta^*(q, a_1 a_2 a_3 \dots a_{n-2} a_{n-1}), a_n) \\ &= \delta^*(\delta^*(\delta^*(q, a_1 a_2 a_3 \dots a_{n-2}), a_{n-1}), a_n) \\ &= \delta^*(\delta^*(\delta^* \dots \delta^*(\delta^*(\delta^*(q, a_1), a_2) a_3) \dots a_{n-2}), a_{n-1}), a_n) \\ &= \delta(\delta(\delta \dots \delta(\delta(\delta(q, a_1), a_2) a_3) \dots a_{n-2}), a_{n-1}), a_n) \\ \delta^*(q, x) &= \delta(\delta(\delta \dots \delta(\delta(\delta(q, a_1), a_2) a_3) \dots a_{n-2}), a_{n-1}), a_n) \\ &= \delta(\delta(\delta \dots \delta(\delta(q_1, a_2) a_3) \dots a_{n-2}), a_{n-1}), a_n) \\ &= \delta(\delta(\delta \dots \delta(q_2, a_3) \dots a_{n-2}), a_{n-1}), a_n) \\ &\quad \vdots \\ &= \delta(\delta(q_{n-2}, a_{n-1}), a_n) \\ &= \delta(q_{n-1}, a_n) \\ &= q'\end{aligned}$$

§ 6.3 有限状态自动机

映射规则的推广

$$\delta : Q \times \Sigma_I \rightarrow Q \quad \text{VS} \quad \delta^* : Q \times \Sigma_I^* \rightarrow Q$$



未对处理完当前符号后，
接下来该处理什么做出
任何规定。



不仅借助于 δ 实现了对单个
输入符号的处理，而且明确
地规定了对一个输入符号串
进行处理的顺序。

有限状态自动机接受的句子和语言

若对于 $x \in \Sigma_I^+$ ，存在 δ^* ，使得 $\delta^*(q_0, x) = p \in F$ 成立，则称 x 能被确定的有限状态自动机 $A_f = (Q, \Sigma_I, \delta, q_0, F)$ 所识别。

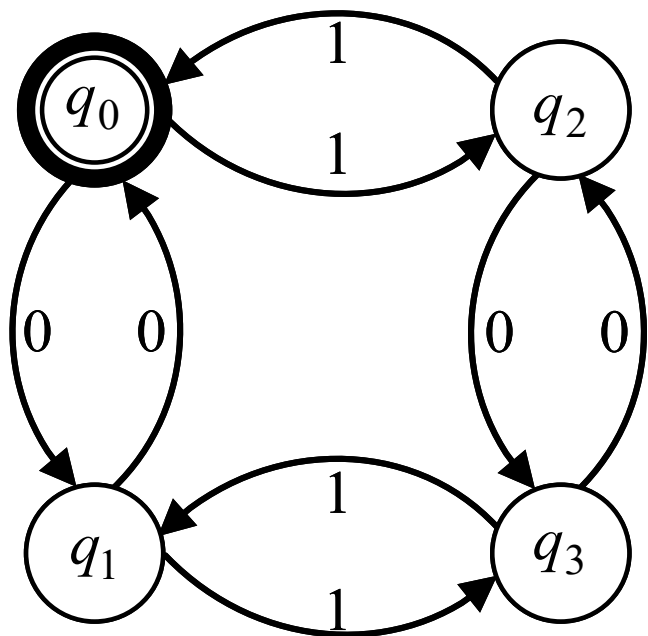
设 $A_f = (Q, \Sigma_I, \delta, q_0, F)$ 是一个确定的有限状态自动机，称

$$L(A_f) = \{x \mid x \in \Sigma_I^+ \text{ 且 } \delta^*(q_0, x) \in F\}$$

为 A_f 所接受的语言。

§ 6.3 有限状态自动机

确定的有限状态自动机接受的语言



从初态 q_0 出发到达 q_1 的符号串
由奇数个**0**和偶数个**1**组成

从初态 q_0 出发到达 q_2 的符号串
由偶数个**0**和奇数个**1**组成

从初态 q_0 出发到达 q_3 的符号串
由奇数个**0**和奇数个**1**组成

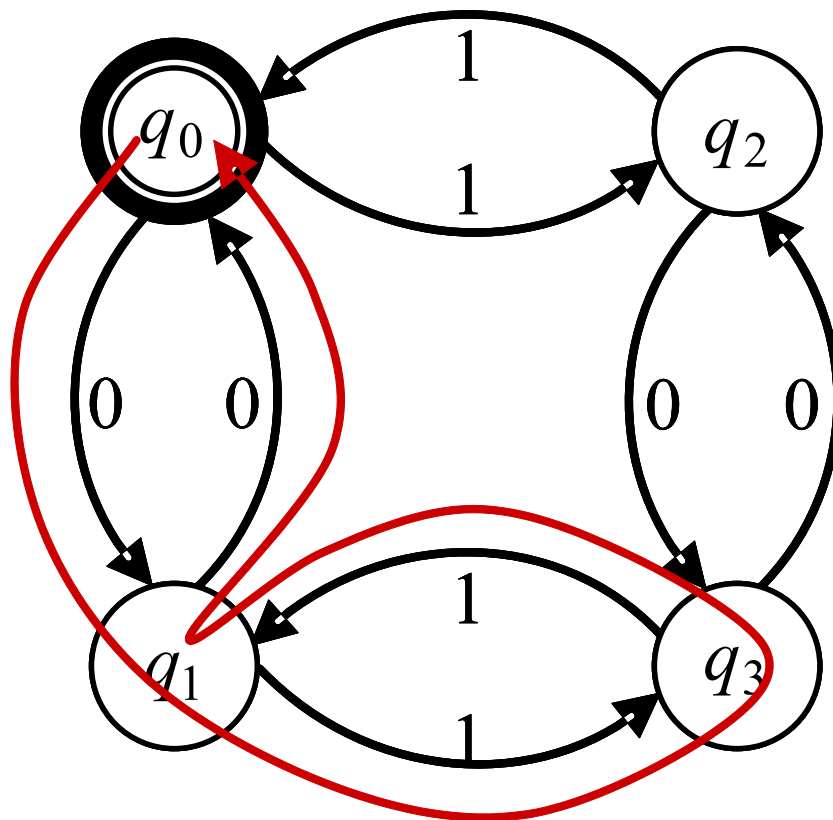
从初态 q_0 出发回到 q_0 的符号串
由偶数个**0**和偶数个**1**组成

由于 $\{0,1\}^+$ 中的符号串能到达的状态必是图示四个状态之一，而终结状态只有 q_0 一个，故图示的自动机只接受由偶数个**0**和偶数个**1**组成的符号串。

§ 6.3 有限状态自动机

确定的有限状态自动机接受的语言

设 $x = 0110$



§ 6.3 有限状态自动机

确定的有限状态自动机的即时描述

$$x = a_1 a_2 a \cdots a_{i-1} a_i a_{i+1} \cdots a_{n-1} a_n$$

当前时刻 A_f 的一个即时描述

$$a_1 a_2 \cdots a_{i-1} q a_i a_{i+1} \cdots a_{n-1} a_n$$

若有 $\delta(q, a_i) = p$, 则下一时刻的即时描述为

$$a_1 a_2 \cdots a_{i-1} a_i p a_{i+1} \cdots a_{n-1} a_n$$

上述过程记为 $a_1 a_2 \cdots a_{i-1} q a_i a_{i+1} \cdots a_{n-1} a_n$

$$\quad \vdash a_1 a_2 \cdots a_{i-1} a_i p a_{i+1} \cdots a_{n-1} a_n$$

.....

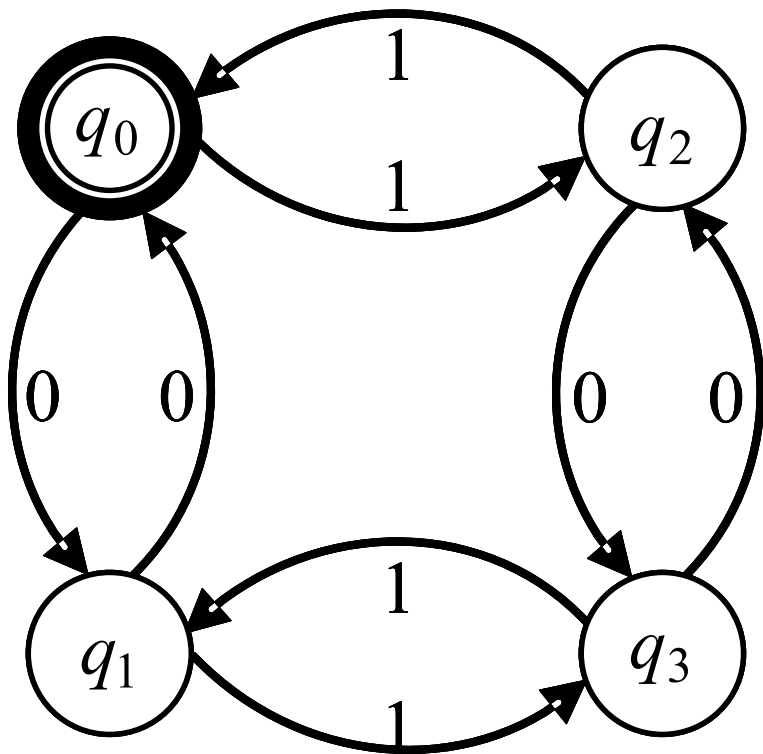
$$\quad \vdash a_1 a_2 \cdots a_{i-1} a_i a_{i+1} \cdots a_{n-1} a_n t$$

若 t 为一个终结状态, 则接受 x , 否则不接受。

§ 6.3 有限状态自动机

确定的有限状态自动机的即时描述

设 $x = 0011$



$q_0 0011 \vdash 0q_1 011$
 $\vdash 00q_0 11$
 $\vdash 001q_2 1$
 $\vdash 0011q_0$

因 q_0 为终结状态, 则接受 x 。

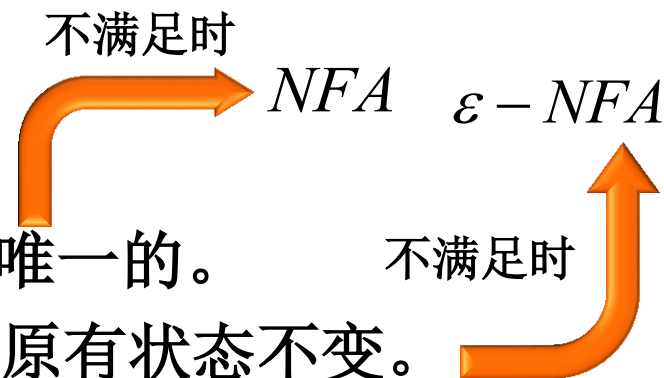
§ 6.3 有限状态自动机

非确定的有限状态自动机

确定的有限状态自动机的特点

1. 每一个映射所到达的新状态都是唯一的。
2. 在未读入任何符号的情况下保持原有状态不变。

上述两个特点至少有一个不满足时, 相应的有限状态自动机成为非确定的有限状态自动机。



$$\delta(q, a) = \{p_1, p_2, \dots, p_m\} \quad \{p_1, p_2, \dots, p_m\} \in 2^Q \quad m > 1$$

$$\delta(q, \lambda) = p \quad p \in \{p_1, p_2, \dots, p_m\} \quad p \neq q$$

§ 6.3 有限状态自动机

非确定的有限状态自动机

一个非确定的有限状态自动机 (**NFA**) 是一个五元式:

$$A_f = (Q, \Sigma_I, \delta, q_0, F)$$

Q 状态的有限集合

Σ_I 输入符号的有限集合, 即字母表

$$\delta : Q \times \Sigma_I \rightarrow 2^Q$$

$q_0 \in Q$ 初始状态

$F \subseteq Q$ 终止状态集合

§ 6.3 有限状态自动机

非确定的有限状态自动机

一个具有 ε 动作的非确定的有限状态自动机 (ε -NFA) 是一个五元式:

$$A_f = (Q, \Sigma_I, \delta, q_0, F)$$

Q 状态的有限集合

Σ_I 输入符号的有限集合, 即字母表

$\delta: Q \times (\Sigma_I \cup \{\lambda\}) \rightarrow 2^Q$ 映射规则

$q_0 \in Q$ 初始状态

$F \subseteq Q$ 终止状态集合

§ 6.3 有限状态自动机

非确定的有限状态自动机

*NFA*情况下推广的映射规则

$$\delta^* : Q \times \Sigma_I^* \rightarrow 2^Q$$

$$\delta^*(q, \lambda) = \{q\}$$

$$\delta^*(q, xa) = \bigcup_{q_i \in \delta^*(q, x)} \delta^*(q_i, a)$$

$$\delta^*(q, a) = \delta(q, a)$$

一个非空的符号串 $x \in \Sigma_I^+$ 能够被一个 *NFA* 所接受是指存在映射规则使 $\delta^*(q_0, x) \cap F \neq \Phi$ 成立。

§ 6.3 有限状态自动机

非确定有限状态自动机的状态转移表表示

$$A_f = (Q, \Sigma_I, \delta, q_0, F)$$

$$Q = \{q_0, q_1, q_2, q_3\} \quad \Sigma_I = \{0, 1\} \quad q_0 \in Q \quad F = \{q_3\}$$

$$\delta(q_0, 0) = \{q_0, q_1\} \quad \delta(q_0, 1) = \{q_0, q_2\} \quad \delta(q_1, 0) = \{q_3\} \quad \delta(q_1, 1) = \Phi$$

$$\delta(q_2, 0) = \Phi \quad \delta(q_2, 1) = \{q_3\} \quad \delta(q_3, 0) = \{q_3\} \quad \delta(q_3, 1) = \{q_3\}$$

在状态名的左上角
打一个小点，表示
初始状态

符号 \ 状态	0	1
$\cdot q_0$	$\{q_0, q_1\}$	$\{\cdot q_0, q_2\}$
q_1	$\{q_3\}$	Φ
q_2	Φ	$\{q_3\}$
q_3	$\{q_3\}$	$\{q_3\}$

在状态名的右下角
打一个小点，表示
终止状态

§ 6.3 有限状态自动机

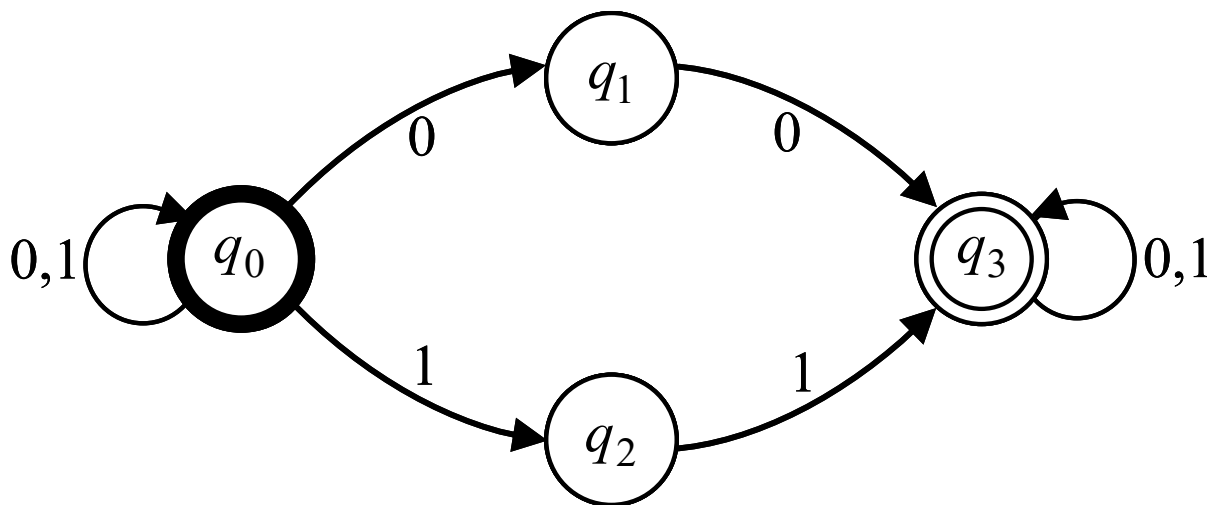
非确定有限状态自动机的状态转移图表示

$$A_f = (Q, \Sigma_I, \delta, q_0, F)$$

$$Q = \{q_0, q_1, q_2, q_3\} \quad \Sigma_I = \{0, 1\} \quad q_0 \in Q \quad F = \{q_3\}$$

$$\delta(q_0, 0) = \{q_0, q_1\} \quad \delta(q_0, 1) = \{q_0, q_2\} \quad \delta(q_1, 0) = \{q_3\} \quad \delta(q_1, 1) = \Phi$$

$$\delta(q_2, 0) = \Phi \quad \delta(q_2, 1) = \{q_3\} \quad \delta(q_3, 0) = \{q_3\} \quad \delta(q_3, 1) = \{q_3\}$$



§ 6.3 有限状态自动机

非确定有限状态自动机的即时描述

$$A_f = (\mathcal{Q}, \Sigma_I, \delta, q_0, F)$$

$$\mathcal{Q} = \{q_0, q_1, q_2, q_3\} \quad \Sigma_I = \{0, 1\} \quad q_0 \in \mathcal{Q} \quad F = \{q_3\}$$

$$\delta(q_0,0) = \{q_0, q_1\} \quad \delta(q_0,1) = \{q_0, q_2\} \quad \delta(q_1,0) = \{q_3\} \quad \delta(q_1,1) = \Phi$$

$$\delta(q_2,0) = \Phi \qquad \delta(q_2,1) = \{q_3\} \qquad \delta(q_3,0) = \{q_3\} \qquad \delta(q_3,1) = \{q_3\}$$

$$q_0 0011 \vdash \begin{cases} 0q_0 011 \vdash \begin{cases} 00q_0 11 \vdash \begin{cases} 001q_0 1 \vdash \begin{cases} 0011q_0 \\ 0011q_2 \end{cases} \\ 001q_2 1 \vdash 0011q_3 \end{cases} \\ 00q_1 11 \rightarrow \text{因 } \delta(q_1, 1) = \Phi, \text{ 故停机。} \end{cases} \\ 0q_1 011 \vdash 00q_3 11 \vdash 001q_3 1 \vdash 0011q_3 \end{cases}$$

§ 6.3 有限状态自动机

同一个字母表上的有限状态自动机之间的等价

A_f 和 A'_f 等价 \longrightarrow 它们接受相同的符号串集合

如何判定两个有限状态自动机之间的等价性呢？

$$A_f = (Q, \Sigma_I, \delta, q_0, F) \iff A'_f = (Q', \Sigma_I, \delta', q'_0, F')$$

$$\Sigma_I = \{a_1, a_2, \dots, a_n\}$$

判定等价与不的步骤

输入为 a_m 时

q_{a_m} A_f 到达的下一个状态

q'_{a_m} A'_f 到达的下一个状态

q_c A_f 的当前状态

q'_c A'_f 的当前状态

(标题行) :

增的
文字;

符 (q_c, q'_c)

以后各列 (第 j 列, $m=j-1$) , 下一状态对标识符;

$$(q_{a_m}, q'_{a_m}), m=1, 2, \dots, n$$

置 $i=1$, 第1行第1列的元素为 (q_0, q'_0) 。

§ 6.3 有限状态自动机

判定等价与否的步骤

2. 确定现行第 i 行中除第1 列之外的所有列单元的内容:

若有 $q_{a_j} \in \delta(q_c, a_j)$ $q'_{a_j} \in \delta'(q'_c, a_j)$

则将 (q_{a_j}, q'_{a_j}) 加入到现行行第 $j+1$ 列单元中。

3. 检查现行行中所有可能的状态对 $(q_{a_j}, q'_{a_j}), j = 1, 2, \dots, n$,

若某个状态对在表的第1列中未出现过, 则将其顺序加到第1列已有状态对的后面;

若某个状态对不满足等价性的要求, 则判 A_f 和 A'_f 不等价, 算法结束。

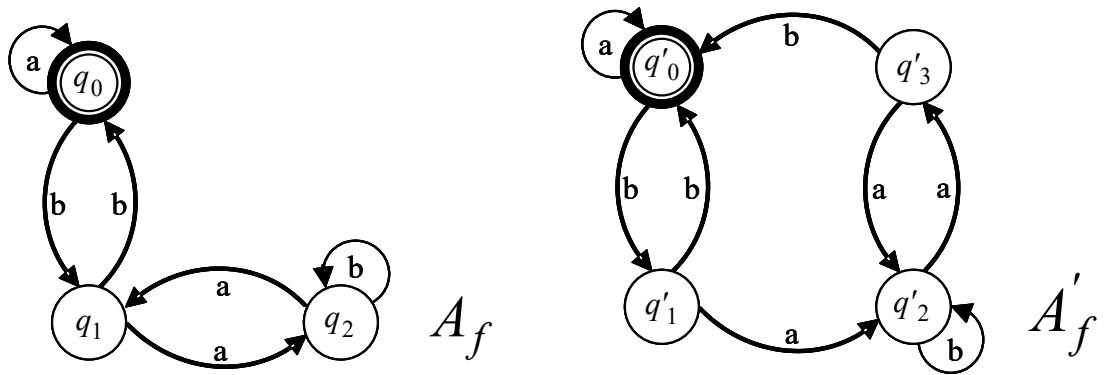
4. 检查第1列中是否还有未处理的状态对。

若有, 令 $i = i+1$, 回到步骤3。

否则, 判 A_f 和 A'_f 等价, 算法结束。

§ 6.3 有限状态自动机

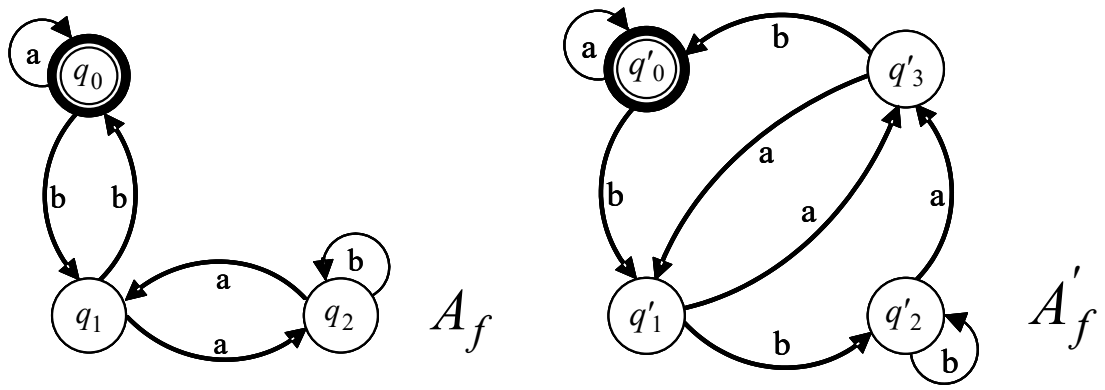
等价性判定举例



状态对 迭代次数			
	(q_c, q'_c)	(q_a, q'_a)	(q_b, q'_b)
1	(q_0, q'_0)	(q_0, q'_0)	(q_1, q'_1)
2	(q_1, q'_1)	(q_2, q'_2)	(q_0, q'_0)
3	(q_2, q'_2)	(q_1, q'_3)	(q_2, q'_2)
4	(q_1, q'_3)	(q_2, q'_2)	(q_0, q'_0)

§ 6.3 有限状态自动机

等价性判定举例



状态对 迭代次数	状态对		
	(q_c, q'_c)	(q_a, q'_a)	(q_b, q'_b)
1	(q_0, q'_0)	(q_0, q'_0)	(q_1, q'_1)
2	(q_1, q'_1)	(q_2, q'_3)	(q_0, q'_2)
3			
4			

§ 6.3 有限状态自动机

DFA与NFA之间的等价性

结论：对于任何一个**DFA**而言，必定可找到一个与其相对应的**NFA**，使两者接受相同的符号串集合。

问题：反过来上述结论成立否？

定理 设 L 是被一个**NFA**接受的语言,则存在一个**DFA**也接受 L 。

§ 6.3 有限状态自动机

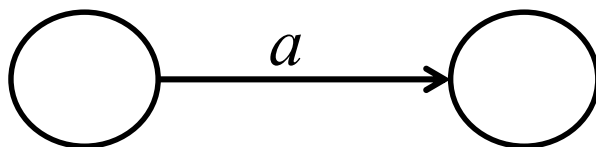
构建与一个**NFA**等价的**DFA**的方法

1. 令 $[q_0]$ 是 A'_f 的初态。
2. 若在 δ 中, 有 $\delta(q, a) = \{p_1, p_2, \dots, p_i\}$,
则在 A'_f 中, 有状态 $[p_1, p_2, \dots, p_i]$ 。
3. 若在 δ 中, 有 $\delta(\{p_1, p_2, \dots, p_i\}, a) = \{r_1, r_2, \dots, r_k\}$,
则在 A'_f 中, 有状态 $[r_1, r_2, \dots, r_k]$ 。
4. 由以上各步骤所确定的状态之间的状态转移由 δ 中相应的映射规则确定。



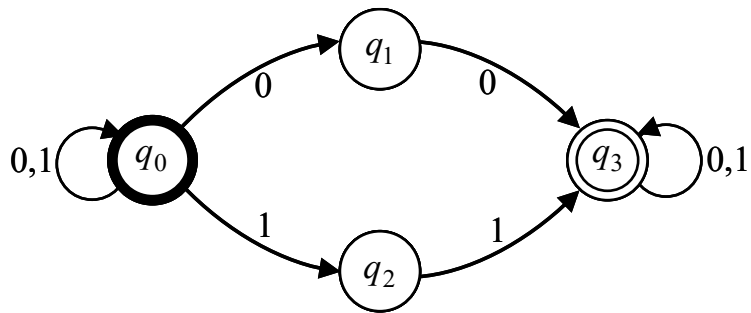
$$\delta(\{p_1, p_2, \dots, p_i\}, a) = \{r_1, r_2, \dots, r_k\}$$

$$[p_1, p_2, \dots, p_i] \quad [r_1, r_2, \dots, r_k]$$



§ 6.3 有限状态自动机

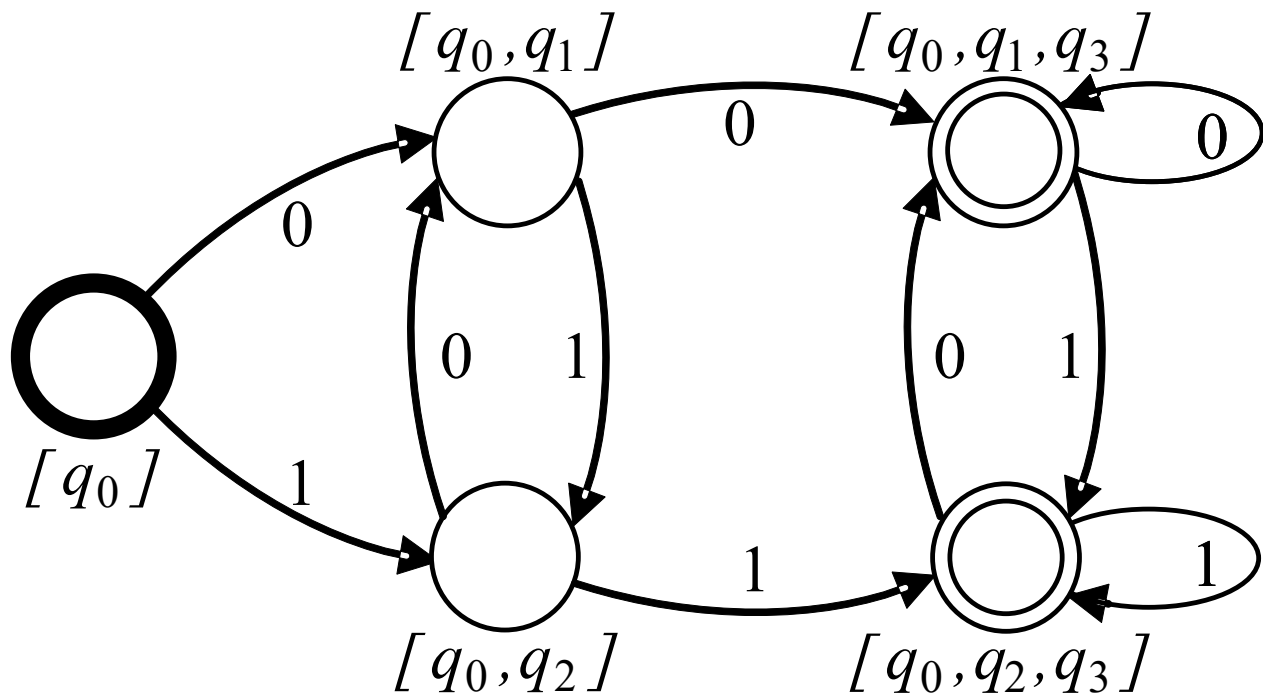
例：求与下述**NFA**等价的**DFA**。



$$A_f = (Q, \Sigma_I, \delta, q_0, F)$$



$$A'_f = (Q', \Sigma_I, \delta', q'_0, F')$$



§ 6.3 有限状态自动机

有限状态文法和有限状态自动机

定理 设文法 $G = (N, T, P, S)$ 是一个有限状态文法, 则必存在一个有限状态自动机 $A_f = (Q, \Sigma_I, \delta, q_0, F)$ 接受该文法所产生的语言, 即 $L(A_f) = L(G)$ 。

定理 设 $A_f = (Q, \Sigma_I, \delta, q_0, F)$ 是一个有限状态自动机, 则必存在一个有限状态文法 $G = (N, T, P, S)$ 产生由该有限状态自动机所接受的语言, 即 $L(G) = L(A_f)$ 。

思考题

教授与农民在火车上相对而坐。

无聊之际，教授对农民说：太无聊了，我们来玩点什么吧。我出一道题，你若不知，给我**5**块钱；你出一道题，我若不知，给你**500**块钱。行不？

农民同意。

教授问：月亮距地球多远？农民一言不发，递给教授**5**元钱。

农民问：上山三条腿，下山四条腿，是什么动物？

教授百思不得其解，无奈递给农民**500**元钱。

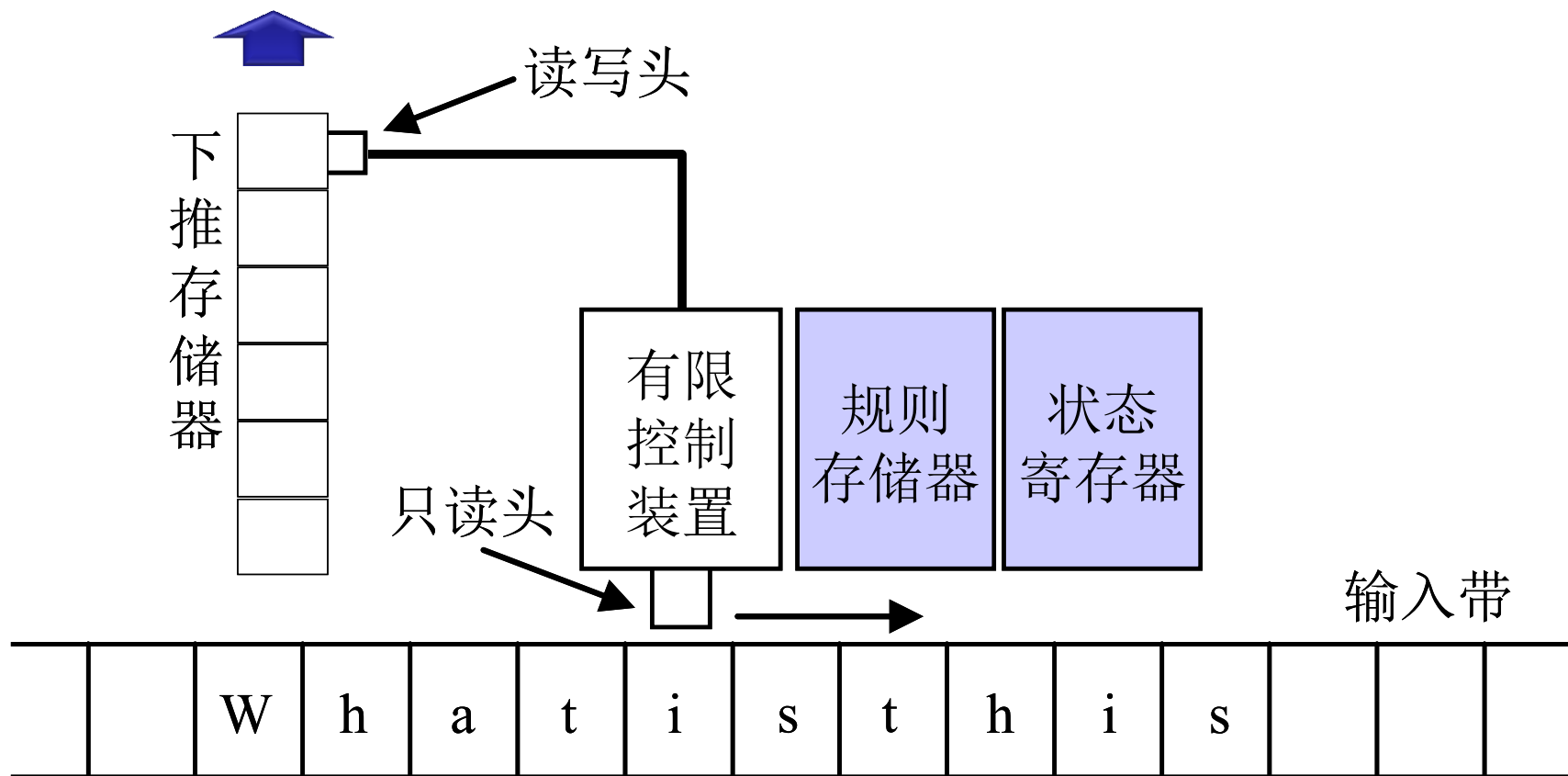
农民接过钱准备睡觉。教授追问：到底是什么动物？

问：农民给出了什么答案？

§ 6.4 下推自动机

Push Down Automaton (PDA)

一个长度不受限制的“后入先出”的堆栈



附加有下推存储器的有限状态自动机！

§ 6.4 下推自动机

定义 一个非确定的下推自动机 (**PDA**) 是一个七元式:

$$A_p = (Q, \Sigma_I, \Gamma, \delta, q_0, Z_0, F)$$

Q 状态的有限集合

Σ_I 输入符号的有限集合, 即字母表

Γ 堆栈符号的有限集合

$q_0 \in Q$ 初始状态

$Z_0 \in \Gamma$ 栈底符号

$F \subseteq Q$ 终止状态集合

$\delta : Q \times (\Sigma_I \cup \{\lambda\}) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$

弹出 Z , 然后将 γ_i 中的符号按照
从右到左的顺序依次压入栈中



$$\delta(q, a, Z) = \{(p_1, \gamma_1), (p_2, \gamma_2), \dots, (p_m, \gamma_m)\}$$



§ 6.4 下推自动机

下推自动机的工作过程

启动时，下推自动机处于初始状态，栈中存放栈底符号，而只读头指向输入符号串的最左一个符号。

进入运行后，下推自动机依据映射规则完成对输入符号串的处理：根据下推自动机的当前状态、当前输入符号或空串以及栈顶的当前堆栈符号，确定下推自动机的下一个状态并改变相应堆栈的内容，同时控制只读头的位置。

随着处理的不断进行，只读头从左到右逐个扫描符号串的每一个符号，直到所有符号均被处理为止。

若扫描完符号串后，下推自动机停留在终结状态集合，或停留在空栈上，则称符号串可为下推自动机所识别。

§ 6.4 下推自动机

下推自动机的即时表示

下推自动机的当前状态

输入符号串的剩余部分

(q, ω, α)

下推自动机当前堆栈中的内容

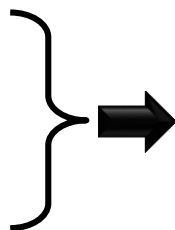
输入串是否被接受的判定:

从下推自动机最初的即时描述出发, 依据当前符号、当前状态、当前堆栈内容和映射规则不断确定下一时刻下推自动机的即时描述, 直至下列情况之一出现:

1. 输入符号串剩余部分为空, 且下推自动机停留在终结状态或停留在空栈上。
2. 输入符号串剩余部分不为空, 但已无合适的映射规则可用。
3. 输入符号串剩余部分为空, 但下推自动机停留的状态不是终结状态, 同时停留的堆栈也非空。



接受



不接受

§ 6.4 下推自动机

下推自动机举例

$$A_p = (Q, \Sigma_I, \Gamma, \delta, q_0, Z_0, F)$$

$$Q = \{q_0, q_1, q_2\} \quad \Sigma_I = \{0, 1\} \quad \Gamma = \{Z_0, A\} \quad F = \{q_2\}$$

1. $\delta(q_0, 0, Z_0) = \{(q_0, AZ_0)\}$
2. $\delta(q_0, 0, A) = \{(q_0, AA)\}$
3. $\delta(q_0, 1, A) = \{(q_1, \lambda)\}$
4. $\delta(q_1, 1, A) = \{(q_1, \lambda)\}$
5. $\delta(q_1, \lambda, Z_0) = \{(q_2, \lambda)\}$

特点:

1. 当待处理当前符号为**0**时，在堆栈中压入**A**。
2. 当待处理当前符号为**1**时，从堆栈中弹出**A**。



$$L(A_p) = \{x \mid x = 0^n 1^n, n \geq 1\}$$

§ 6.4 下推自动机

下推自动机举例

$$A_p = (Q, \Sigma_I, \Gamma, \delta, q_0, Z_0, F)$$

1. $\delta(q_0, 0, Z_0) = \{(q_0, AZ_0)\}$
2. $\delta(q_0, 0, A) = \{(q_0, AA)\}$
3. $\delta(q_0, 1, A) = \{(q_1, \lambda)\}$
4. $\delta(q_1, 1, A) = \{(q_1, \lambda)\}$
5. $\delta(q_1, \lambda, Z_0) = \{(q_2, \lambda)\}$

当输入符号串为“**010101**”时

$$\begin{array}{ccc} (q_0, 010101, Z_0) & \xrightarrow{(1)} & (q_0, 10101, AZ_0) \\ & \xrightarrow{(3)} & (q_1, 0101, Z_0) \end{array}$$

因此后无合适映射规则可用, 故判断不可接受。

§ 6.4 下推自动机

下推自动机举例

$$A_p = (Q, \Sigma_I, \Gamma, \delta, q_0, Z_0, F) \quad (q_0, 000111, Z_0) \quad \xrightarrow{(1)} (q_0, 00111, AZ_0)$$

$$1. \delta(q_0, 0, Z_0) = \{(q_0, AZ_0)\} \quad \xrightarrow{(2)} (q_0, 0111, AAZ_0)$$

$$2. \delta(q_0, 0, A) = \{(q_0, AA)\} \quad \xrightarrow{(2)} (q_0, 111, AAAZ_0)$$


$$3. \delta(q_0, 1, A) = \{(q_1, \lambda)\}$$

$$4. \delta(q_1, 1, A) = \{(q_1, \lambda)\} \quad \xrightarrow{(3)} (q_1, 11, AAZ_0)$$

$$5. \delta(q_1, \lambda, Z_0) = \{(q_2, \lambda)\} \quad \xrightarrow{(4)} (q_1, 1, AZ_0)$$

当输入符号串为“**000111**”时

$$\xrightarrow{(4)} (q_1, \lambda, Z_0)$$

 因停留在空栈上, 故判断可接受。

$$\xrightarrow{(5)} (q_2, \lambda, \lambda)$$

§ 6.4 下推自动机

下推自动机课堂练习题

$$A_p = (Q, \Sigma_I, \Gamma, \delta, q_0, Z_0, F)$$


$$Q = \{q_0\} \quad \Sigma_I = \{a, b, c, d\} \quad \Gamma = \{S, A, B, C, D\} \quad Z_0 = S \quad F = \Phi$$

$$1. \delta(q_0, c, S) = \{(q_0, DAB), (q_0, C)\} \quad 2. \delta(q_0, a, D) = \{(q_0, \lambda)\}$$

$$3. \delta(q_0, d, C) = \{(q_0, \lambda)\} \quad 4. \delta(q_0, b, B) = \{(q_0, \lambda)\}$$

$$5. \delta(q_0, a, A) = \{(q_0, AB), (q_0, CB)\}$$

问：符号串“**cadbb**”可否被接受？


$$N(A_p) = \{x \mid x = ca^n db^n, n \geq 0\}$$

§ 6.4 下推自动机

上下文无关文法和下推自动机

下推自动机接受的语言类和上下文无关文法产生的语言类是一致的。

文法的规范表示

- 乔姆斯基范式文法 (*CNFG*)

$$A \rightarrow BC \quad \text{或} \quad A \rightarrow a$$

定理 任何上下文无关语言都可以由乔姆斯基范式文法所生成。
证明:

只要证 $A \rightarrow \beta$ 可由 $A \rightarrow BC$ 或 $A \rightarrow a$ 形式的产生式表示即可。

根据定义, $A \rightarrow \beta$

$$A \in N \quad \beta \in \Sigma^+ = (N \cup T)^+$$

对 $A \rightarrow \beta$ 做如下处理:

设其右端由 n ($n \geq 1$) 个符号 (终结符或非终结符) 所组成

$$A \rightarrow X_1 X_2 \cdots X_n \quad X_i \in N \cup T, \quad i = 1, 2, \dots, n$$

§ 6.4 下推自动机

上下文无关文法和下推自动机

证明（续）：

若 $X_i, i = 1, 2, \dots, n$ 为终结符，则引入非终结符 X'_i ，并用

$$A \rightarrow X_1 X_2 \cdots X'_i \cdots X_n$$

$$X'_i \rightarrow X_i$$

替换原产生式。

这样，可得到如下所示的一组新产生式，

$$A \rightarrow X_1 X_2 \cdots X_n \quad X_i \in N, i = 1, 2, \dots, n$$

$$A \rightarrow a \quad a \in T$$

对右端仅包含单一非终结符的产生式，用相关的产生式组进行置换，直至右端由 2 个以上非终结符构成为止。

至此，所有产生式均具有以下形式：

$$A \rightarrow X_1 X_2 \cdots X_n \quad X_i \in N, i = 1, 2, \dots, n \quad n \geq 2$$

$$A \rightarrow a \quad a \in T$$

§ 6.4 下推自动机

上下文无关文法和下推自动机
证明（续）：

对所有形如 $A \rightarrow X_1 X_2 \cdots X_n$ 的产生式作如下分解

$$A \rightarrow X_1 X_1'$$

$$X_1' \rightarrow X_2 X_2'$$

$$X_2' \rightarrow X_3 X_3'$$

$$\vdots$$

$$X_{n-2}' \rightarrow X_{n-1} X_n$$

至此，所有新旧产生式组均具有以下形式：

$$A \rightarrow BC \quad A, B, C \in N$$

$$A \rightarrow a \quad a \in T$$

证毕。

§ 6.4 下推自动机

上下文无关文法和下推自动机
文法的规范表示

- 格雷巴赫范式文法 (*GNFG*)

$$A \rightarrow a\gamma \quad \gamma \in N^*$$



$$A \rightarrow aA_1A_2\cdots A_n \quad n \geq 1$$

$$A \rightarrow a$$

定理 任何上下文无关语言都可以由格雷巴赫范式文法生成。


定理 对于任意的**PDA** A_p 而言, 均存在上下文无关文法**GNFG**, 使 $L(GNFG) = N(A_p)$ 。

§ 6.4 下推自动机

上下文无关文法和下推自动机 自嵌入性质

$$A \xRightarrow[G]{*} \alpha_1 A \alpha_2 \quad \alpha_1, \alpha_2 \in \Sigma^+$$

$$1. S \rightarrow uAy \quad 2. A \rightarrow vAx \quad 3. A \rightarrow w$$


$$S \xRightarrow{(1)} uAy \xRightarrow{(2)} uvAxy \xRightarrow{(2)} uv^2Ax^2y \xRightarrow{(2)} \cdots \xRightarrow{(2)} uv^iAx^iy \xRightarrow{(3)} uv^iwx^iy$$

结论：若一个上下文无关文法是非自嵌入的，则由它所产生的语言是一个有限状态语言，可以由有限状态自动机所识别；否则，由它所产生的语言是一个严格的上下文无关语言。一个严格的上下文无关语言，仅能够用下推自动机进行识别。

§ 6.4 下推自动机

确定的和非确定的下推自动机之间的关系

确定的和非确定的下推自动机不等价。



反例：



不能被任何确定的下推自动机所接受



$$L(G) = \{xx^R \mid x \in \{0,1\}^*\}$$

可以被下述非确定的下推自动机所接受

$$A_p = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \Phi)$$

$$Q = \{q_1, q_2\} \quad \Sigma = \{0, 1\} \quad \Gamma = \{A, B, C\} \quad Z_0 = A$$

- | | |
|--|--|
| 1. $\delta(q_1, 0, A) = \{(q_1, BA)\}$ | 2. $\delta(q_1, 1, C) = \{(q_1, CC), (q_2, \lambda)\}$ |
| 3. $\delta(q_1, 1, A) = \{(q_1, CA)\}$ | 4. $\delta(q_2, 0, B) = \{(q_2, \lambda)\}$ |
| 5. $\delta(q_1, 0, B) = \{(q_1, BB), (q_2, \lambda)\}$ | 6. $\delta(q_2, 1, C) = \{(q_2, \lambda)\}$ |
| 7. $\delta(q_1, 0, C) = \{(q_1, BC)\}$ | 8. $\delta(q_1, \lambda, A) = \{(q_2, \lambda)\}$ |
| 9. $\delta(q_1, 1, B) = \{(q_1, CB)\}$ | 10. $\delta(q_2, \lambda, A) = \{(q_2, \lambda)\}$ |

§ 6.4 下推自动机

确定的和非确定的下推自动机之间的关系

确定的和非确定的下推自动机不等价。

1. $\delta(q_1, 0, A) = \{(q_1, BA)\}$

$$2. \delta(q_1, 1, C) = \{(q_1, CC), (q_2, \lambda)\}$$

$$3. \delta(q_1, 1, A) = \{(q_1, CA)\}$$

4. $\delta(q_2, 0, B) = \{(q_2, \lambda)\}$

$$5. \delta(q_1, 0, B) = \{(q_1, BB), (q_2, \lambda)\}$$

$$6. \delta(q_2, 1, C) = \{(q_2, \lambda)\}$$

$$7. \delta(q_1, 0, C) = \{(q_1, BC)\}$$

8. $\delta(q_1, \lambda, A) = \{(q_2, \lambda)\}$

9. $\delta(q_1, 1, B) = \{(q_1, CB)\}$

10. $\delta(q_2, \lambda, A) = \{(q_2, \lambda)\}$

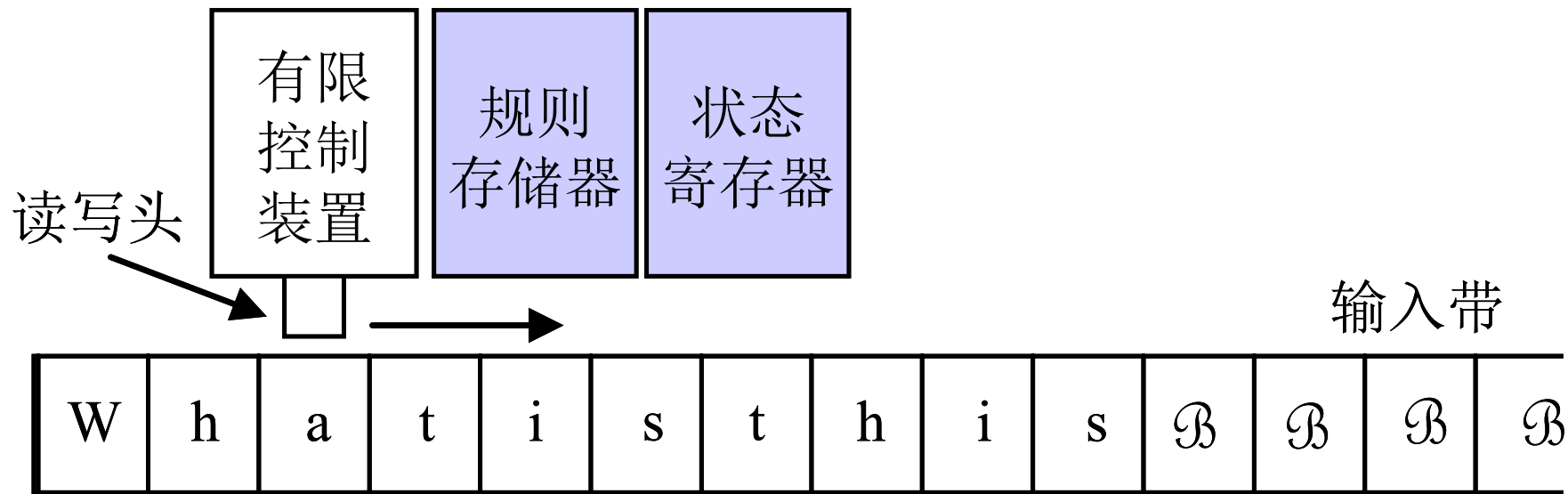
Diagram illustrating two vertical red lines. The left line is labeled $x00x^R$ and the right line is labeled $x11x^R$.

$$z00y00y^R00z^R$$

对称中心

§ 6.5 图灵机

Turing Machine (TM)



- 带读写头并可左右移动的有限状态自动机！
- 有限状态自动机和扩展的下推自动机的合体！

§ 6.5 图灵机

定义 一个确定的图灵机 (TM) 是一个七元式:

$$A_T = (Q, \Sigma_I, \mathfrak{B}, \Gamma, \delta, q_0, F)$$



基本图灵机

Q 状态的有限集合

Γ 可出现在输入带上的带符号集合

$\Sigma_I \subseteq \Gamma - \{\mathfrak{B}\}$ 输入符号的有限集合

\mathfrak{B} 空白符号

$q_0 \in Q$ 初始状态

$F \subseteq Q$ 终止状态集合

$\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$



$\delta(q, X) = (p, Y, L)$ 或 $\delta(q, X) = (p, Y, R)$

§ 6.5 图灵机

定义 一个非确定的图灵机 (TM) 是一个七元式:

$$A_T = (Q, \Sigma_I, \mathfrak{B}, \Gamma, \delta, q_0, F)$$

Q 状态的有限集合

Γ 可出现在输入带上的带符号集合

$\Sigma_I \subseteq \Gamma - \{\mathfrak{B}\}$ 输入符号的有限集合

\mathfrak{B} 空白符号

$q_0 \in Q$ 初始状态

$F \subseteq Q$ 终止状态集合

$\delta(q, X) = \{(p_1, Y_1, D_1), (p_2, Y_2, D_2), \dots, (p_n, Y_n, D_n)\}$

$D_i \in \{L, R\}, i = 1, 2, \dots, n$



 非确定的图灵机和基本图灵机是等价的。

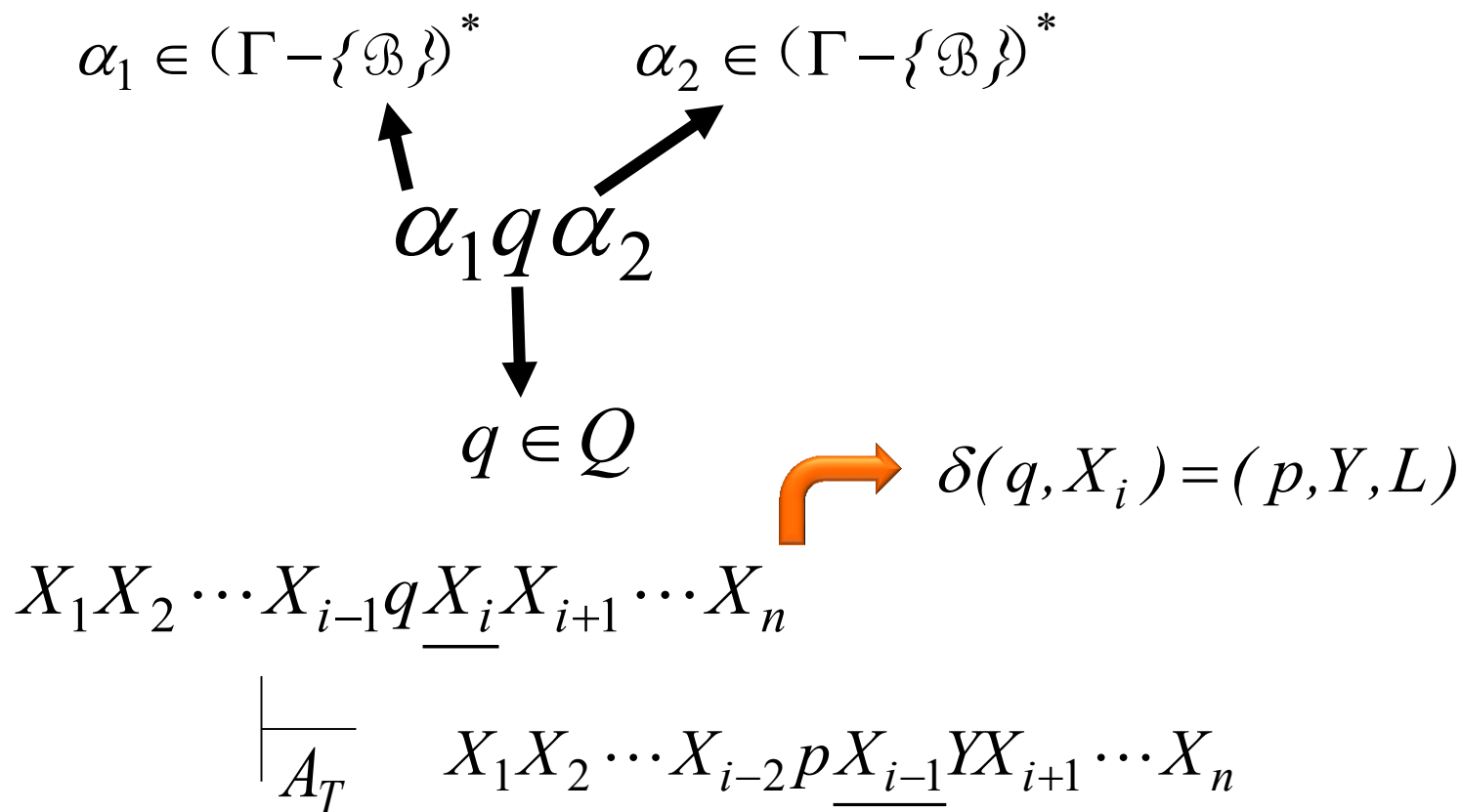
§ 6.5 图灵机

图灵机的即时描述

其最左符号为读写头正注视的符号

当前时刻位于读写头右边的带上符号

当前时刻位于读写头左边的带上符号



§ 6.5 图灵机

图灵机和有限状态自动机的不同点

1. 图灵机即时描述中的状态符可随读写头向左、向右双向移动；有限状态自动机即时描述中的状态符只能单向向右移动。
2. 对图灵机而言，一旦其状态符变为终结状态符，则立即可以对输入符号串做出是否接受的判断；有限状态自动机需要扫描完整整个输入符号串才能对做出判断。
3. 图灵机即时描述中的符号串可为除空白符号之外的任何带符号；有限状态自动机即时描述中的符号串来自于字母表。

图灵机的工作过程


启动后，图灵机根据当前状态和当前带上符号，由映射规则对当前带上符号进行改写并控制读写头向左或向右移动一个单元。上述操作不断进行直至图灵机到达下列事态之一：到达状态为终结状态，或者到达状态虽为非终结状态，但后续已无映射规则可用。


§ 6.5 图灵机

图灵机举例 $A_T = (Q, \Sigma_I, \mathcal{B}, \Gamma, \delta, q_0, F)$

$$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\} \quad \Sigma_I = \{0, 1\} \quad \Gamma = \{0, 1, X, Y, \mathcal{B}\} \quad F = \{q_5\}$$

1. $\delta(q_0, 0) = (q_1, X, R)$
2. $\delta(q_1, 0) = (q_1, 0, R)$
3. $\delta(q_2, Y) = (q_2, Y, L)$
4. $\delta(q_3, Y) = (q_3, Y, R)$
5. $\delta(q_4, 0) = (q_4, 0, L)$
6. $\delta(q_1, Y) = (q_1, Y, R)$
7. $\delta(q_2, X) = (q_3, X, R)$
8. $\delta(q_3, \mathcal{B}) = (q_5, Y, 0)$
9. $\delta(q_4, X) = (q_0, X, R)$
10. $\delta(q_1, 1) = (q_2, Y, L)$
11. $\delta(q_2, 0) = (q_4, 0, L)$

 $\{x \mid x = 0^n 1^n, n \geq 1\}$

输入: **0011**  $q_0 0011$

$\begin{array}{c} (1) \\ \hline \end{array}$	$Xq_1 011$	$\begin{array}{c} (2) \\ \hline \end{array}$	$X0q_1 11$	$\begin{array}{c} (10) \\ \hline \end{array}$	$Xq_2 0Y1$	$\begin{array}{c} (11) \\ \hline \end{array}$	$q_4 X0Y1$
$\begin{array}{c} (9) \\ \hline \end{array}$	$Xq_0 0Y1$	$\begin{array}{c} (1) \\ \hline \end{array}$	$XXq_1 Y1$	$\begin{array}{c} (6) \\ \hline \end{array}$	$XXYq_1 1$	$\begin{array}{c} (10) \\ \hline \end{array}$	$XXq_2 YY$
$\begin{array}{c} (3) \\ \hline \end{array}$	$Xq_2 XYY$	$\begin{array}{c} (7) \\ \hline \end{array}$	$XXq_3 YY$	$\begin{array}{c} (4) \\ \hline \end{array}$	$XXYq_3 Y$	$\begin{array}{c} (4) \\ \hline \end{array}$	$XXYYq_3$
$\begin{array}{c} (8) \\ \hline \end{array}$	$XXYYq_5$						

§ 6.5 图灵机

无约束文法与图灵机

与有限状态自动机和正则文法之间、下推自动机和上下文无关文法之间所具有的等价关系类似，图灵机接受的语言类和无约束文法产生的语言类也是一致的。

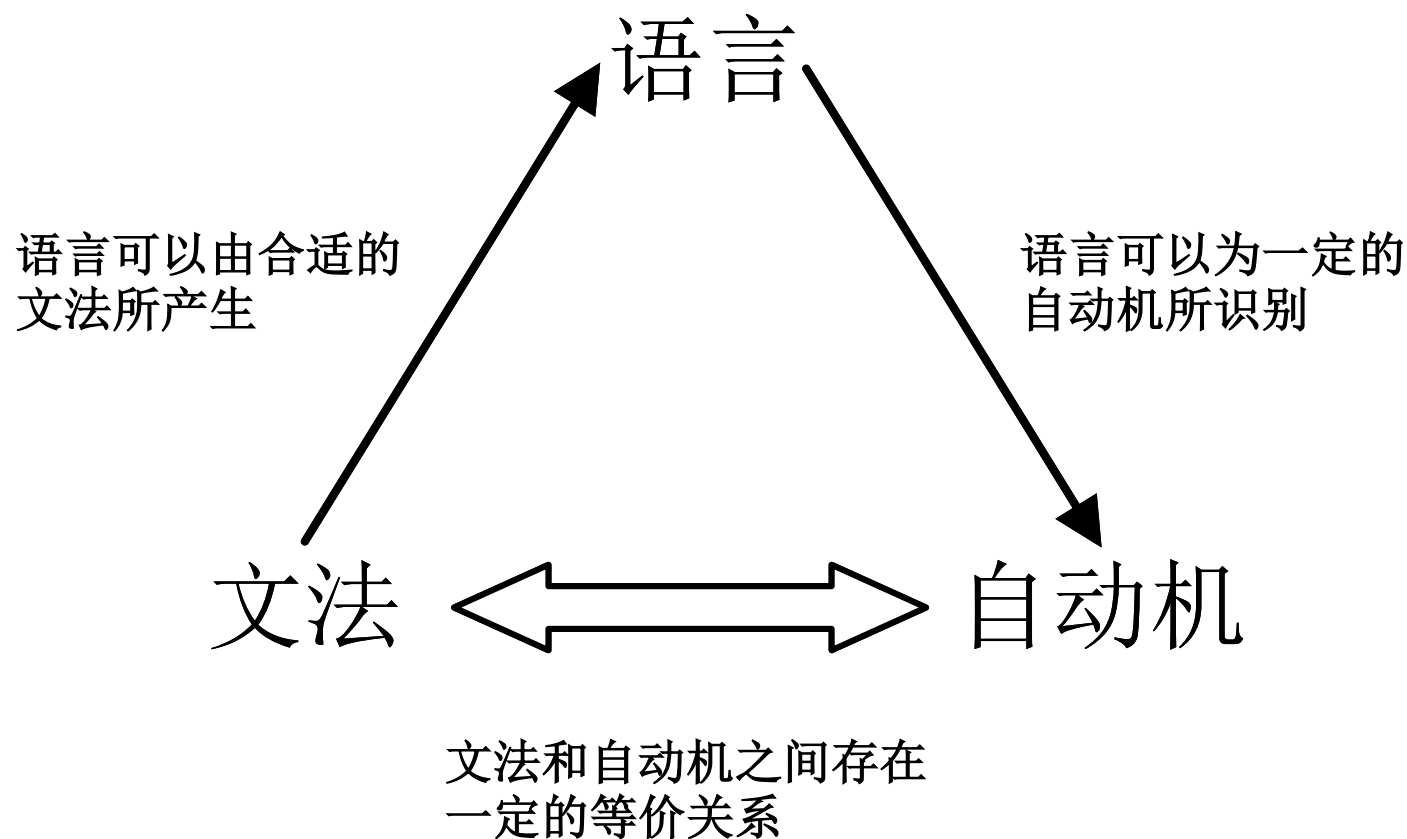
定理 设文法 $G = (N, T, P, S)$ 是一个短语结构文法， $L(G)$ 为由 G 所产生的语言，则存在一个图灵机 A_T ，使 $L(A_T) = L(G)$ 。

定理 设 $L(A_T)$ 是一个图灵机 A_T 所接受的语言，则必存在一个短语结构文法 $G = (N, T, P, S)$ ，使 $L(G) = L(A_T)$ 。

。

§ 6.6 关于语言、文法和自动机的再讨论

语言、文法和自动机三者之间的关系



§ 6.6 关于语言、文法和自动机的再讨论

语言的命名

从例子谈起 $L = \{x \mid x = 0^n 1^n, n \geq 1\}$

$$G = (N, T, P, S)$$

$$N = \{S, A\}$$

$$T = \{0, 1\}$$

$$P: (1) S \rightarrow 0A1$$

$$(2) A \rightarrow 0A1$$

$$(3) A \rightarrow \lambda$$

$$G = (N, T, P, S)$$

$$N = \{S, A\}$$

$$T = \{0, 1\}$$

$$P: (1) S \rightarrow 0S1$$

$$(2) S \rightarrow 01$$

短语结构文法

上下文无关文法

规定：将一个可由多个类型的文法所产生的语言命名为由受约束最多的文法所产生的语言。

§ 6.6 关于语言、文法和自动机的再讨论

语言类型的确定

确定语言类型很重要：

选择正确，可以对症下药。
选择错误，则会无功而返。

➡ 非常希望能够确定一个语言的类型。

可否确定呢？

如果不能完全确定的话，可以确定到什么程度呢？

$$L = \{x \mid x = 0^n 1^n, n \geq 1\}$$

在很多情况下确定一个语言不是什么语言更有实际意义。

§ 6.6 关于语言、文法和自动机的再讨论

语言类型的确定

$$L = \{x \mid x = 0^n 1^n, n \geq 1\}$$

语言的特点：

- 句子具有某种“对称性”
- 句子中所含符号0和1的个数相等
- 符号0出现在符号1之前

➡ 为了识别该语言中的句子，必须记住已读入的子串中所包含符号0的个数。

符号0的个数可以是无穷。



无法用一个有限状态自动机来实现。

§ 6.6 关于语言、文法和自动机的再讨论

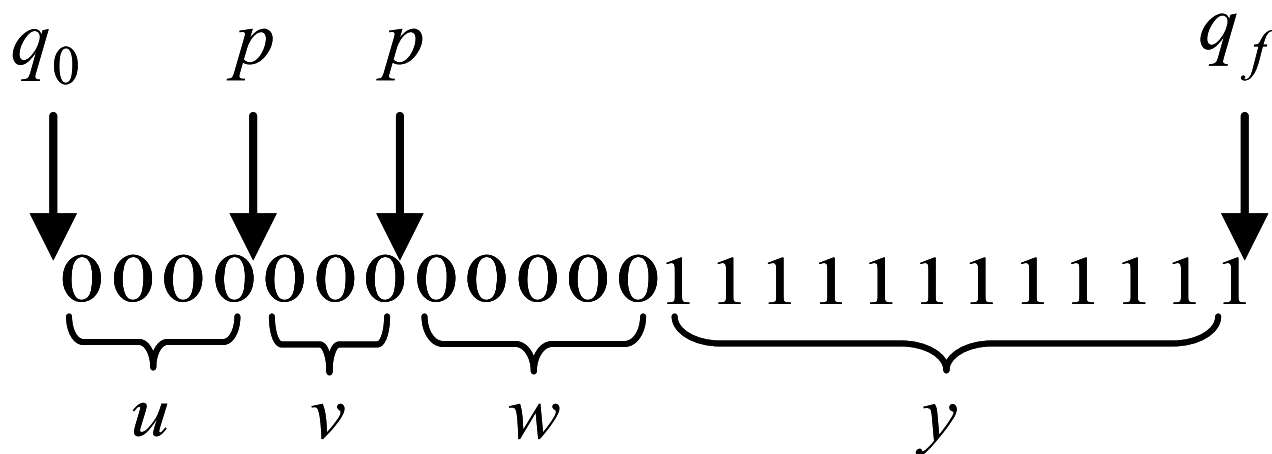
语言类型的确定

$L = \{x \mid x = 0^n 1^n, n \geq 1\}$ 无法用有限状态自动机来识别。

证明（反证法）

设存在一个确定的有限状态自动机 A_f 可识别 L 。其中 A_f 的状态数为 k 。

则对于 $0^n 1^n, n > k$ ，必有



➡ $uvw y \in L \quad |uvw| = |y| = n$

§ 6.6 关于语言、文法和自动机的再讨论

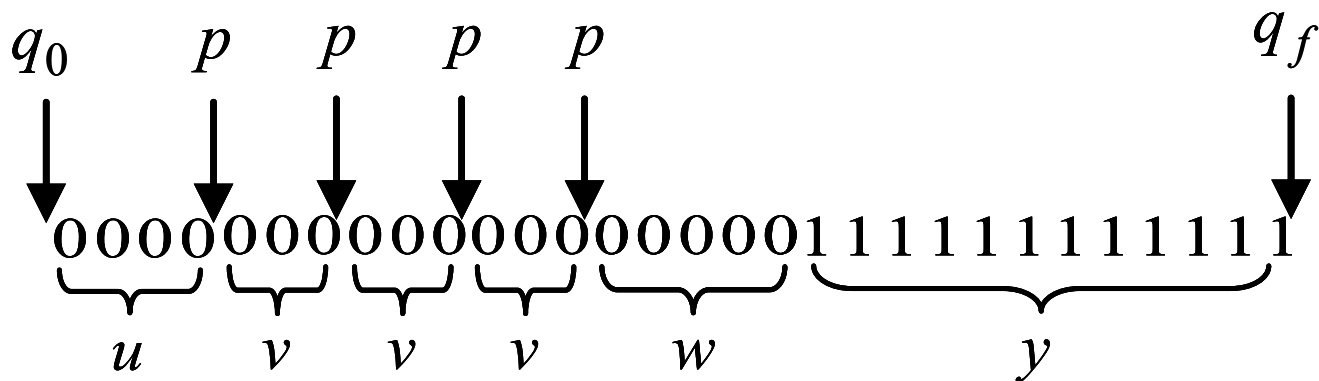
语言类型的确定

$L = \{x \mid x = 0^n 1^n, n \geq 1\}$ 无法用有限状态自动机来识别。

证明（续）

$$\begin{aligned}\delta(q_0, uvwy) &= \delta(\delta(q_0, u), vwy) = \delta(p, vwy) \\ &= \delta(\delta(p, v), wy) = \delta(p, wy) = q_f\end{aligned}$$

$$uv^i wy, i \geq 2 \rightarrow A_f$$

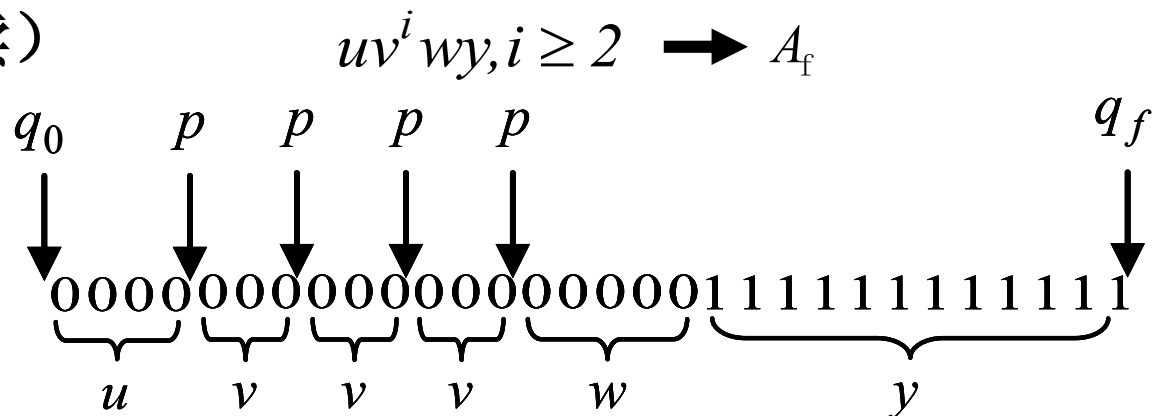


§ 6.6 关于语言、文法和自动机的再讨论

语言类型的确定

$L = \{x \mid x = 0^n 1^n, n \geq 1\}$ 无法用有限状态自动机来识别。

证明 (续)



$$\begin{aligned}\delta(q_0, uv^i wy) &= \delta(\delta(q_0, u), v^i wy) = \delta(p, v^i wy) \\ &= \delta(\delta(p, v), v^{i-1} wy) = \delta(p, v^{i-1} wy) \\ &= \delta(\delta(p, v), v^{i-2} wy) = \delta(p, v^{i-2} wy) \\ &= \dots = \delta(p, vwy) \\ &= \delta(\delta(p, v), wy) = \delta(p, wy) = q_f\end{aligned}$$

➡ $uv^i wy, i \geq 2$ 也能被识别。但是, $|uv^i w| > |uvw| = |y|, i \geq 2$ 。矛盾!

§ 6.6 关于语言、文法和自动机的再讨论

语言类型的确定

正则语言的泵引理 设 L 是一个正则语言,则存在（依赖于 L 的）正整数 k ,使得对于任何 $y_1xy_2 \in L$, 只要 $|x| \geq k$, 就一定存在非空的 v 和 $x = uvw$, 使对于任何 $i \geq 0$, 都有 $y_1uv^iwy_2 \in L$ 。

定理的直观解释：给定一个属于正则语言 L 的句子，任意截取它的一个子串，只要该子串的长度足够长，就一定可以从中找到非空的子串 v 使得删除（也称泵出） v 或任意多次膨胀（也称泵入） v 所得到的新句子仍然属于 L 。

§ 6.6 关于语言、文法和自动机的再讨论

语言类型的确定


定理 设 L 是一个符号串的集合, 若对于所有的正整数 k , 存在 $y_1xy_2 \in L, |x| \geq k$, 且对于满足 $x = uvw$ 的任意非空的 v , 存在某个选定的 $i \geq 0$, 使有 $y_1uv^iwy_2 \notin L$, 则称 L 不是一个正则语言。

例: $L = \{a^l b^m b^n, l > m \geq 1, n \geq 1\}$ 不是一个正则语言。

§ 6.6 关于语言、文法和自动机的再讨论

语言类型的确定

上下文无关语言的泵引理 设 L 是一个上下文无关语言,则存在正整数 k ,使得对于任何 $z \in L$, 只要 $|z| \geq k$, 就可将 z 表示为 $z = uvwxy$, 且对于任何 $i \geq 0$, 都有 $uv^iwx^iy \in L$ 。

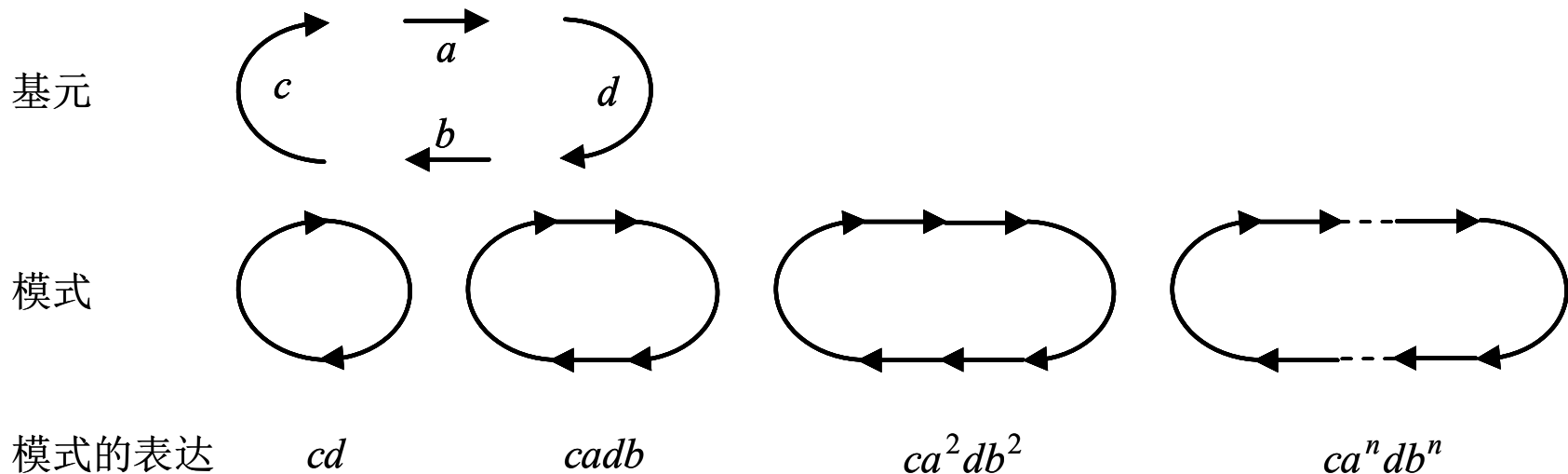

$$|vx| \geq 1 \quad |vwx| \leq k$$

定理 设 L 是一个符号串的集合,若对于所有的正整数 k ,存在 $z \in L$, $|z| \geq k$, 且 $z = uvwxy$, 对满足 $|vx| \geq 1$ 和 $|vwx| \leq k$ 的 z 的任意划分, 存在某个选定的 $i \geq 0$, 使有 $uv^iwx^iy \notin L$, 则称 L 不是一个上下文无关语言。

§ 6.6 关于语言、文法和自动机的再讨论

从语言构建自动机

问题：图形模式识别



对上述图形模式进行归纳整理，可知其可由下述语言所描述：

$$L = \{ x \mid x = ca^n db^n, n \geq 0 \}$$

§ 6.6 关于语言、文法和自动机的再讨论

从语言构建自动机
所构建的自动机

$$A_p = (Q, \Sigma_I, \Gamma, \delta, q_0, Z_0, F)$$

$$Q = \{q_0, q_1, q_2\} \quad \Sigma_I = \{a, b, c, d\} \quad \Gamma = \{Z_0, A, C, \} \quad F = \Phi$$

- P:**
1. $\delta(q_0, c, Z_0) = \{(q_1, CZ_0)\}$ — 发现头符号 c 。
 2. $\delta(q_1, a, C) = \{(q_1, AC)\}$ — 在头符号 c 后发现 a ;
 3. $\delta(q_1, a, A) = \{(q_1, AA)\}$ — 在符号 a 后发现 a ;
 4. $\delta(q_1, d, A) = \{(q_2, A)\}$ — 在符号 a 后发现 d ;
 5. $\delta(q_1, d, C) = \{(q_2, \lambda)\}$ — 在头符号 c 后发现 d ;
 6. $\delta(q_2, b, A) = \{(q_2, \lambda)\}$ — 发现符号 b ;
 7. $\delta(q_2, \lambda, C) = \{(q_2, \lambda)\}$ — 符号 a 和符号 b 的个数相等;
 8. $\delta(q_2, \lambda, Z_0) = \{(q_0, \lambda)\}$ — 句子符合要求。

§ 6.7 句法分析

句法分析：对输入模式进行剖析，确定其所属类别，并给出其结构描述。

{ 基于文法的句法分析
基于自动机的句法分析

给定一个文法 $G = (N, T, P, S)$ 和一个符号串 x ，如果能由起始符 S 出发，通过适用文法中的产生式最终导出给定的符号串 x ，则说 x 属于给定文法所对应的模式类，否则说不属于给定文法所对应的模式类。

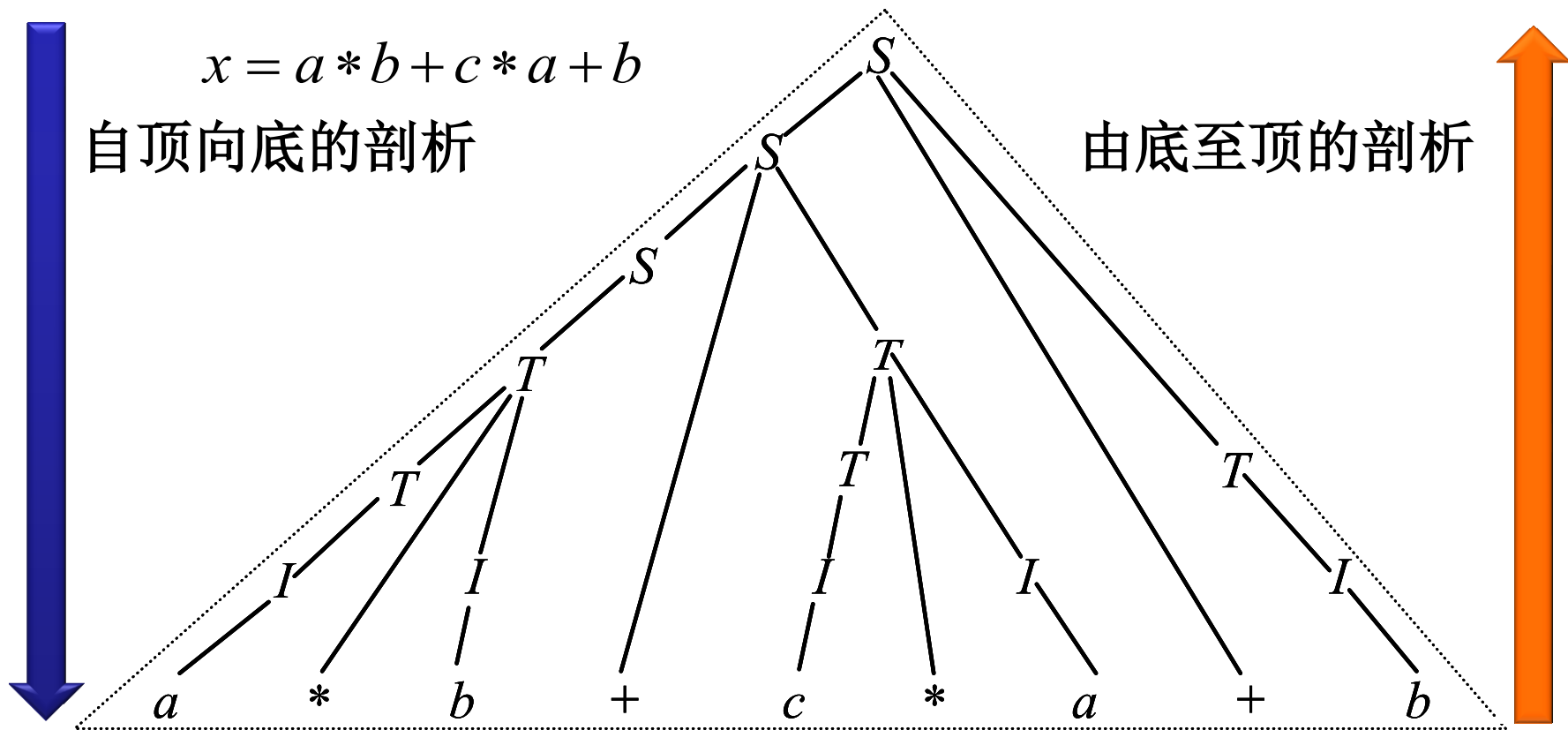
上下文无关文法  用导出树表示

§ 6.7 句法分析

例 $G = (N, T, P, S)$

$$N = \{S, T\} \quad t = \{a, b, c, +, *\}$$

P: 1. $S \rightarrow T$ 2. $S \rightarrow S + T$ 3. $T \rightarrow I$ 4. $T \rightarrow T * I$
5. $I \rightarrow a$ 6. $I \rightarrow b$ 7. $I \rightarrow c$



§ 6.7 句法分析

自顶向底的剖析（正向剖析）

搜索+回溯

$$S \rightarrow X_1 X_2 \cdots X_n$$



若为非终结符, 则建立子目标。

$$X_2 \rightarrow Y_1 Y_2 \cdots Y_m$$

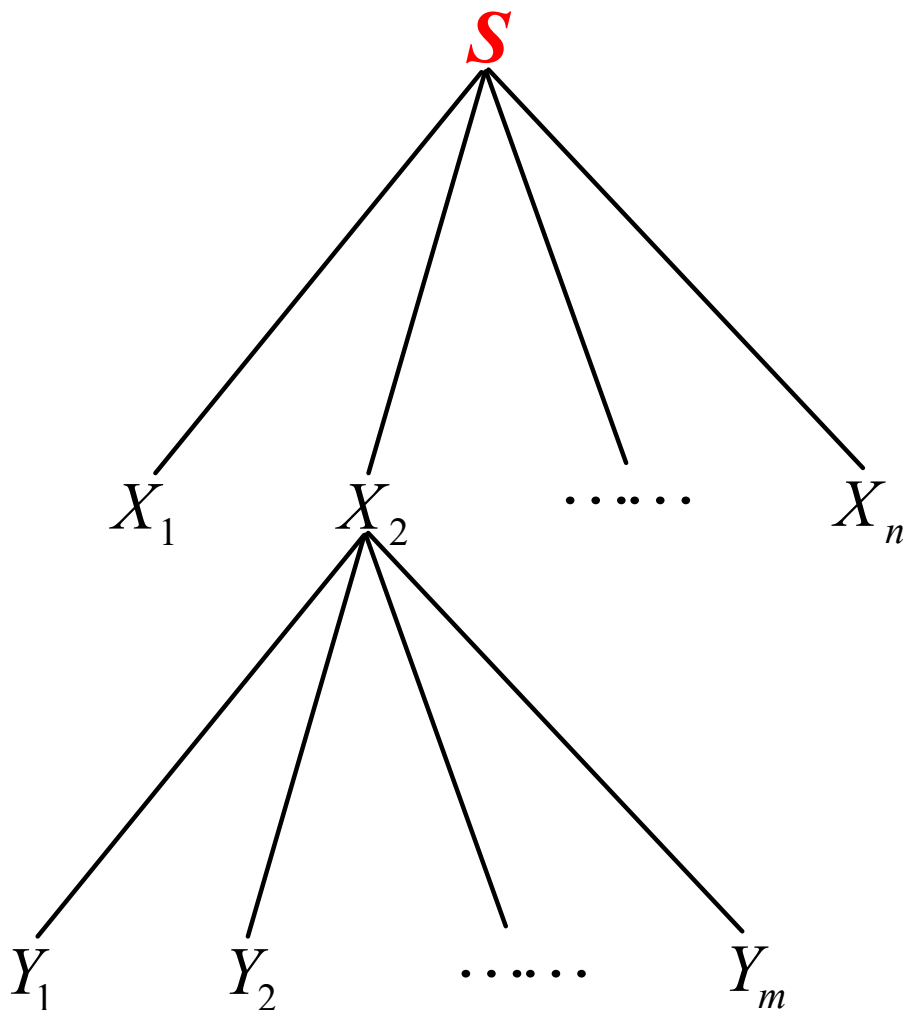
.....

直到所有的非终结符
被终结符所置换。

检查所生成的符号串
是否与输入串一致。

若一致, 结束;

否则, 做回溯处理。



§ 6.7 句法分析

正向剖析的最终判断由下列事态决定：

1. 某次导出的符号串与输入符号串相匹配。 ➡ 可导出
2. 尝试所有选择之后，仍未导出输入符号串。 ➡ 不可导出

反向剖析的工作过程与正向剖析正好相反。

从输入符号串开始，不断地反向适用产生式进行置换处理。

反向剖析的最终判断由下列事态决定：

1. 最后一次置换后到达起始符。 ➡ 可导出
2. 未到达起始符，且不再有可能的置换发生。 ➡ 不可导出

剖析方法的缺点：盲目性大，效率低下

§ 6.7 句法分析

基于三角表格的反向剖析算法（CYK算法）

适用对象：以乔姆斯基范式形式表示的上下文无关文法

设待识别的输入符号串为 $x = a_1a_2 \cdots a_n$

1. 单个符号的派生

$A_{i,1}, 1 \leq i \leq n$ 可派生 a_i 的非终结符

寻找 $A_{i,1} \rightarrow a_i, 1 \leq i \leq n$, 若有, 记录 $A_{i,1}$, 若无, 记 $A_{i,1} = \phi$ 。

2. 两个符号组成的子串的派生

$A_{i,2}, 1 \leq i \leq n-1$ 可派生 $x_{i,2} = a_i a_{i+1}, 1 \leq i \leq n-1$ 的非终结符

寻找 $A_{i,2} \rightarrow A_{i,1} A_{i+1,1}$, 若有, 记录 $A_{i,2}$, 若无, 记 $A_{i,2} = \phi$ 。

3. 三个符号组成的子串的派生

$A_{i,3}, 1 \leq i \leq n-2$ 可派生 $x_{i,3} = a_i a_{i+1} a_{i+2}, 1 \leq i \leq n-2$ 的非终结符

寻找 $A_{i,3} \rightarrow A_{i,1} A_{i+1,2}$, 若有, 记录 $A_{i,3}$, 若无, 记 $A_{i,3} = \phi$ 。
 $A_{i,3} \rightarrow A_{i,2} A_{i+2,1}$

.....

§ 6.7 句法分析

基于三角表格的反向剖析算法（CYK算法）

.....

n. 输入符号串 x 的派生

$A_{1,n}$ 可派生 $x = a_1 a_2 \cdots a_n$ 的非终结符

寻找 $\left\{ \begin{array}{l} A_{1,n} \rightarrow A_{1,1} A_{2,n-1} \\ A_{1,n} \rightarrow A_{1,2} A_{3,n-2} \\ \dots\dots\dots \\ A_{1,n} \rightarrow A_{1,n-1} A_{n,1} \end{array} \right.$, 若有, 记录 $A_{1,n}$, 若无, 记 $A_{1,n} = \phi$ 。

最终判决: 若 $A_{1,n}$ 的可能取值中包含 S ,
则表示 $x \in L(G)$, 否则 $x \notin L(G)$ 。

}


§ 6.7 句法分析

基于三角表格的反向剖析算法（CYK算法）

$$A_{i,j} \quad x_{i,j} = a_i a_{i+1} \cdots a_{i+j-1}, 1 \leq i \leq n - j + 1$$

 起始于 i 、长度为 j 的子串

检查有无 $A_{i,j} \rightarrow A_{i,k} A_{i+k,j-k}, \quad 1 \leq k \leq j-1$

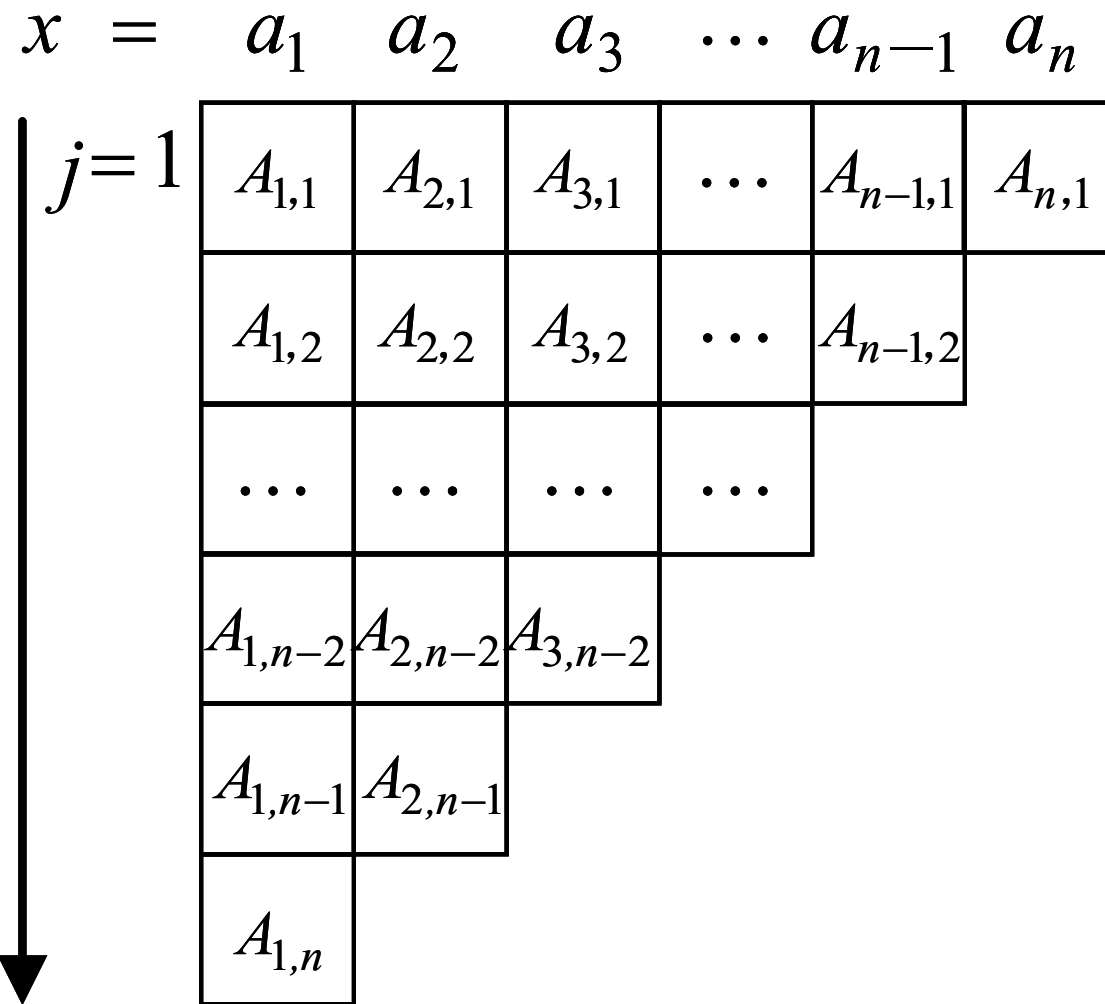
 $A_{i,k} \neq \phi$
 $A_{i+k,j-k} \neq \phi$

若有，记录 $A_{i,j}$ ；

若无，记 $A_{i,j} = \phi$ 。

§ 6.7 句法分析

基于三角表格的反向剖析算法（CYK算法）



§ 6.7 句法分析

CYK算法举例

$$G = (N, T, P, S)$$

$$N = \{S, T\} \quad t = \{a, b, c, +, *\}$$

P: 1. $S \rightarrow T$ 2. $S \rightarrow S + T$ 3. $T \rightarrow I$ 4. $T \rightarrow T * I$
5. $I \rightarrow a$ 6. $I \rightarrow b$ 7. $I \rightarrow c$

问: $x = a + b * c$ 能否由 G 所生成?

解: 1. 将产生式改写成乔姆斯基范式形式

$$G' = (N', T, P', S)$$

$$N' = \{S, T, A, B, M\}$$

P: 1. $S \rightarrow SB$ 2. $B \rightarrow AT$ 3. $T \rightarrow TC$ 4. $C \rightarrow MI$
5. $S \rightarrow a$ 6. $S \rightarrow b$ 7. $S \rightarrow c$ 8. $A \rightarrow +$
9. $T \rightarrow a$ 10. $T \rightarrow b$ 11. $T \rightarrow c$ 12. $M \rightarrow *$
13. $I \rightarrow a$ 14. $I \rightarrow b$ 15. $I \rightarrow c$

§ 6.7 句法分析

CYK算法举例

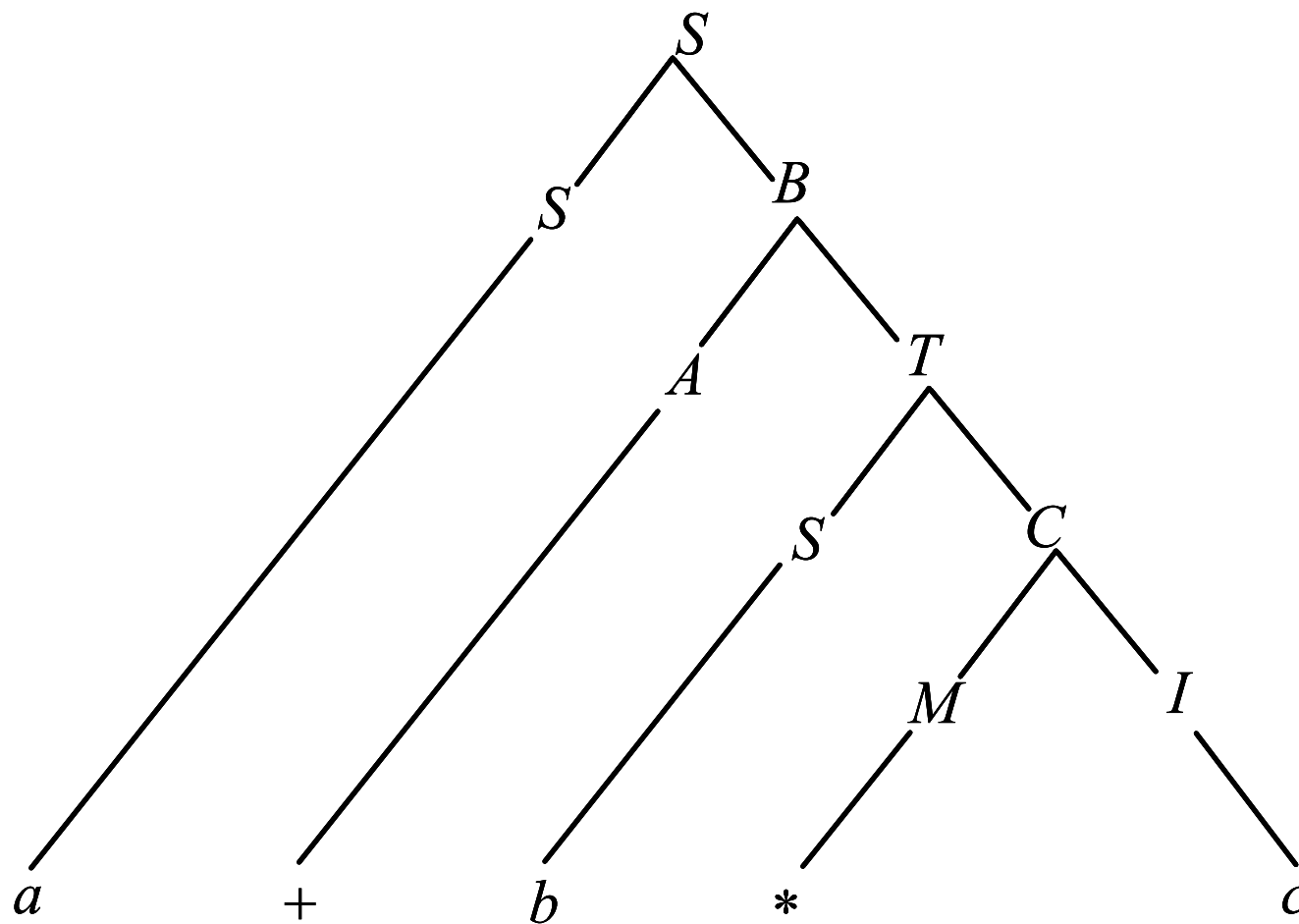
- P:** 1. $S \rightarrow SB$ 2. $B \rightarrow AT$ 3. $T \rightarrow TC$ 4. $C \rightarrow MI$
5. $S \rightarrow a$ 6. $S \rightarrow b$ 7. $S \rightarrow c$ 8. $A \rightarrow +$
9. $T \rightarrow a$ 10. $T \rightarrow b$ 11. $T \rightarrow c$ 12. $M \rightarrow *$
13. $I \rightarrow a$ 14. $I \rightarrow b$ 15. $I \rightarrow c$

2. 构造三角表格

$x =$	a	$+$	b	$*$	c
$j = 1$	S, T, I	A	S, T, I	M	S, T, I
	ϕ	B	ϕ	C	
	S	ϕ	T		
	ϕ	B			
	S				

§ 6.7 句法分析

CYK算法举例



$S \rightarrow SB$
 $\rightarrow SAT$
 $\rightarrow SASC$
 $\rightarrow SASMI$
 $\rightarrow aASMI$
 $\rightarrow a + SMI$
 $\rightarrow a + bMI$
 $\rightarrow a + b * I$
 $\rightarrow a + b * c$

§ 6.8 文法推断

给定的句子集合

➡ 推断相应的文法

$$G = (N, T, P, S)$$

$$L(G) \quad \overline{L(G)} = T^* - L(G)$$

$$R^+ = \{x \mid x \in L(G)\} \quad \text{正样本集合}$$

$$R^- = \{x \mid x \in \overline{L(G)}\} \quad \text{负样本集合}$$

➡ 根据上述不完全样本集合，不一定能得到“理想”的文法推断结果。

➡ 文法推断是一个需要“智力”的工作。

§ 6.8 文法推断

正则文法的推断

$$R^+ = \{X_1, X_2, \dots, X_m\} \Rightarrow G = (N, T, P, S)$$

推断步骤

1. 列出 R^+ 中的所有终结符，令其为终结符集合。
2. 对 R^+ 中的符号串 $X_i, i=1, 2, \dots, m$ ，适当引入非终结符，构建恰好能产生 X_i 的产生式组。
3. 对所得文法中的非终结符和产生式组，进行必要的化简和合并，最终得到所推断的文法。

$$X_i = a_{i1}a_{i2} \cdots a_{in_i}$$

$$S \rightarrow a_{i1}Z_{i1} \quad Z_{i1} \rightarrow a_{i2}Z_{i2}$$


$$\dots\dots\dots Z_{in_i-2} \rightarrow a_{in_i-1}Z_{in_i-1} \quad Z_{in_i-1} \rightarrow a_{in_i}$$


§ 6.8 文法推断


正则文法的推断


例 $R^+ = \{01, 001, 0001\}$  $G = (N, T, P, S)$

解 $T = \{0, 1\}$

$X_1 = 01$  $S \rightarrow 0Z_{11} \quad Z_{11} \rightarrow 1$

$X_2 = 001$  $S \rightarrow 0Z_{21} \quad Z_{21} \rightarrow 0Z_{22} \quad Z_{22} \rightarrow 1$

$X_3 = 0001$  $S \rightarrow 0Z_{31} \quad Z_{31} \rightarrow 0Z_{32} \quad Z_{32} \rightarrow 0Z_{33} \quad Z_{33} \rightarrow 1$

 $G = (N, T, P, S)$

$N = \{S, Z_{11}, Z_{21}, Z_{22}, Z_{31}, Z_{32}, Z_{33}\} \quad T = \{0, 1\}$

$P: S \rightarrow 0Z_{11} \quad Z_{11} \rightarrow 1$

$S \rightarrow 0Z_{21} \quad Z_{21} \rightarrow 0Z_{22} \quad Z_{22} \rightarrow 1$

$S \rightarrow 0Z_{31} \quad Z_{31} \rightarrow 0Z_{32} \quad Z_{32} \rightarrow 0Z_{33} \quad Z_{33} \rightarrow 1$

§ 6.8 文法推断

正则文法的推断

例 $R^+ = \{01, 001, 0001\}$ $\Rightarrow G = (N, T, P, S)$

$\Rightarrow G = (N, T, P, S)$

$N = \{S, Z_{11}, Z_{21}, Z_{22}, Z_{31}, Z_{32}, Z_{33}\}$ $T = \{0, 1\}$

$P:$ $S \rightarrow 0Z_{11}$ $Z_{11} \rightarrow 1$
 $S \rightarrow 0Z_{21}$ $Z_{21} \rightarrow 0Z_{22}$ $Z_{22} \rightarrow 1$
 $S \rightarrow 0Z_{31}$ $Z_{31} \rightarrow 0Z_{32}$ $Z_{32} \rightarrow 0Z_{33}$ $Z_{33} \rightarrow 1$

化简 $G = (N, T, P, S)$

$N = \{S, A_1, A_2\}$ $T = \{0, 1\}$

$P:$ $S \rightarrow 0A_1$ $A_1 \rightarrow 1$

~~$S \rightarrow 0A_1$ $A_1 \rightarrow 0A_2$ $A_2 \rightarrow 1$~~

~~$S \rightarrow 0A_1$ $A_1 \rightarrow 0A_2$ $A_2 \rightarrow 0A_2$ $A_2 \rightarrow 1$~~

§ 6.8 文法推断

正则文法的推断

例 $R^+ = \{01, 001, 0001\}$ $\Rightarrow G = (N, T, P, S)$

$\Rightarrow G = (N, T, P, S)$

$$N = \{S, A_1, A_2\} \quad T = \{0, 1\}$$

$$P: S \rightarrow 0A_1 \quad A_1 \rightarrow 1$$

$$A_1 \rightarrow 0A_2 \quad A_2 \rightarrow 1 \quad A_2 \rightarrow 0A_2$$

进一步化简

$$G = (N, T, P, S)$$

$$N = \{S, A_1\} \quad T = \{0, 1\}$$

$$P: S \rightarrow 0A_1 \quad A_1 \rightarrow 0A_1 \quad A_1 \rightarrow 1$$

$$\Rightarrow L(G) = \{0^n 1 \mid n = 1, 2, \dots\}$$

§ 6.8 文法推断


正则文法的推断：余码文法

定义 考虑链码（由符号串接而成）集合 A ，称

$$\{x \mid \alpha x \in A\}$$

为 A 舍去 α 后的余码集合，记为 $D_\alpha A$ 。

$$\alpha x \in A \quad \xrightarrow{\text{蓝色箭头}} \quad D_\alpha(\alpha x) = x$$

 形式微商算子

若 $\alpha = \alpha_1 \alpha_2$ ，则 $D_\alpha A = D_{\alpha_1 \alpha_2} A = D_{\alpha_2}(D_{\alpha_1} A)$

若 $\alpha = \alpha_1 \alpha_2 \cdots \alpha_n$ ，则 $D_\alpha A = D_{\alpha_1 \alpha_2 \cdots \alpha_n} A = D_{\alpha_n}(\cdots D_{\alpha_2}(D_{\alpha_1} A) \cdots)$

§ 6.8 文法推断

正则文法的推断：余码文法

$$R^+ \rightarrow G_c = (N_c, T_c, P_c, S_c)$$

步骤：

<1> 检查 R^+ ，用其中所包含的所有互异的终结符形成终结符集合 T_c 。

<2> 根据形式微商算子的运算规则，求出 R^+ 所有可能的余码集合，记 $S_c = U_1 = D_\lambda R^+$ ，并用 U_2, \dots, U_n 标记其余的余码集合，则 $N_c = \{S_c, U_2, \dots, U_n\}$ 。

<3-1> 若 $D_a U_i = U_j, i, j = 1, 2, \dots, n$ ，则 $U_i \rightarrow aU_j$ 。

<3-2> 若 $D_a U_i = \{\lambda\}, i = 1, 2, \dots, n$ ，则 $U_i \rightarrow a$ 。

§ 6.8 文法推断

正则文法的推断： K 余文法

定义 考虑链码（由符号串接而成）集合 A ，称

$$\{x \mid \alpha x \in A, |x| \leq K\}$$

为 A 舍去 α 后的 K 余码集合，记为 $D_{\alpha}^K A$ 。



K 余文法 $G_K = (N_K, T_K, P_K, S_K)$

§ 6.8 文法推断

正则文法的推断： K 余文法

例 $R^+ = \{01, 001, 0001\}$ $\Rightarrow G_3 = (N_3, T_3, P_3, S_3)$

$D_\alpha \backslash K$	4	3	2	1
$D_\lambda R^+$	$\{01, 001, 0001\}$	$\{01, 001\}$	$\{01\}$	Φ
$D_0 R^+$	$\{1, 01, 001\}$	$\{1, 01, 001\}$	$\{1, 01\}$	$\{1\}$
$D_{00} R^+ = D_0(D_0 R^+)$	$\{1, 01\}$	$\{1, 01\}$	$\{1, 01\}$	$\{1\}$
$D_{01} R^+ = D_1(D_0 R^+)$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$
$D_{000} R^+ = D_0(D_{00} R^+)$	$\{1\}$	$\{1\}$	$\{1\}$	$\{1\}$
$D_{001} R^+ = D_1(D_{00} R^+)$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$
$D_{0001} R^+ = D_1(D_{000} R^+)$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$

§ 6.8 文法推断

正则文法的推断：K余文法

$D_\alpha \backslash K$	4	3	2	1
$D_\lambda R^+$	$\{01, 001, 0001\}$	$\{01, 001\}$	$\{01\}$	Φ
$D_0 R^+$	$\{1, 01, 001\}$	$\{1, 01, 001\}$	$\{1, 01\}$	$\{1\}$
$D_{00} R^+ = D_0(D_0 R^+)$	$\{1, 01\}$	$\{1, 01\}$	$\{1, 01\}$	$\{1\}$
$D_{01} R^+ = D_1(D_0 R^+)$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$
$D_{000} R^+ = D_0(D_{00} R^+)$	$\{1\}$	$\{1\}$	$\{1\}$	$\{1\}$
$D_{001} R^+ = D_1(D_{00} R^+)$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$
$D_{0001} R^+ = D_1(D_{000} R^+)$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$	$\{\lambda\}$

$$G_3 = (N_3, T_3, P_3, S_3)$$

$$T_3 = \{0, 1\} \quad N_3 = \{S_3, U_2, U_3, U_4\}$$

$$S_3 = U_1 = \{01, 001\} \quad U_2 = \{1, 01, 001\}$$

$$U_3 = \{1, 01\} \quad U_4 = \{1\}$$

$$P_3: \quad S_3 \rightarrow 0U_2 \quad U_2 \rightarrow 0U_3 \quad U_2 \rightarrow 1$$

$$U_3 \rightarrow 0U_4 \quad U_3 \rightarrow 1 \quad U_4 \rightarrow 1$$

§ 6.8 文法推断

上下文无关文法的推断

启发式的方法+试探性的步骤

● 自嵌套性质的推断

- ✓ 对给定正样本集合中的每一个符号串，试探性地删除其中的一些子串，并询问用户，余下的子串是否是可接受的。
- ✓ 如果余下的子串是可接受的，则将删除的子串重复若干次后在原位置处插入形成一个新的串，并再次询问用户，新生成的串是否是可接受的。如果仍是可接受的，则判断待求文法的产生式中存在递归结构。

§ 6.8 文法推断

上下文无关文法的推断

- 自嵌套性质的推断

uwx

若 w 和 u^iwx^i 均可接受, 则推断有产生式

$$A \rightarrow uAx$$

$$A \rightarrow w$$

§ 6.8 文法推断

上下文无关文法的推断

例 $R^+ = \{cd, cadb, caadbb, caaadbbb, caaaaadbbbbb, caaaaaadbbbbbb\}$

➡ $G = (N, T, P, S)$

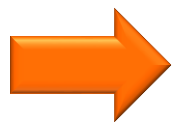
从最短的符号串开始

➡ $S \rightarrow AB \quad A \rightarrow c \quad B \rightarrow d$

接着考察次短符号串

➡ $B \rightarrow aBb \quad B \rightarrow d$

$$L = \{x \mid x = ca^n db^n, n \geq 0\}$$



$$G = (N, T, P, S)$$

$$T = \{a, b, c, d\} \quad N = \{S, A, B\}$$

$$P: S \rightarrow AB \quad B \rightarrow aBb$$

$$A \rightarrow c \quad B \rightarrow d$$



§ 6.8 文法推断

上下文无关文法的推断

- 基于分割子模式的推断

- ✓ 对给定正样本进行分析和归纳，将每个样本分割成若干个性质相同或相似的部分，并将它们组合在一起，形成若干子模式样本集。
- ✓ 对每个子模式样本集进行文法推断，得到相应的文法。
- ✓ 对所有子模式的推断文法进行综合，最终形成所需文法。

§ 6.8 文法推断

上下文无关文法的推断

- 基于样本结构的推断

$$R^+ = \{a + a, a + a + a, a + a + a + a, a + a + a + a + a, a + a + a + a + a + a\}$$

不难发现，该样本集合中的样本呈现出以下规律：
除集合中的第一个样本之外，其余的任一个样本均可
由前一个样本在其后链接符号串“+ a ”得到。



$$G = (N, T, P, S)$$

$$T = \{a, +\} \quad N = \{S, A\}$$

$$P: S \rightarrow A + a \quad A \rightarrow A + a$$

$$A \rightarrow a$$



若干图片材料取自
网络，特此致谢。





谢谢聆听!



中国科学技术大学
University of Science and Technology of China



中国科学技术大学