第5节 预备知识

内容概览

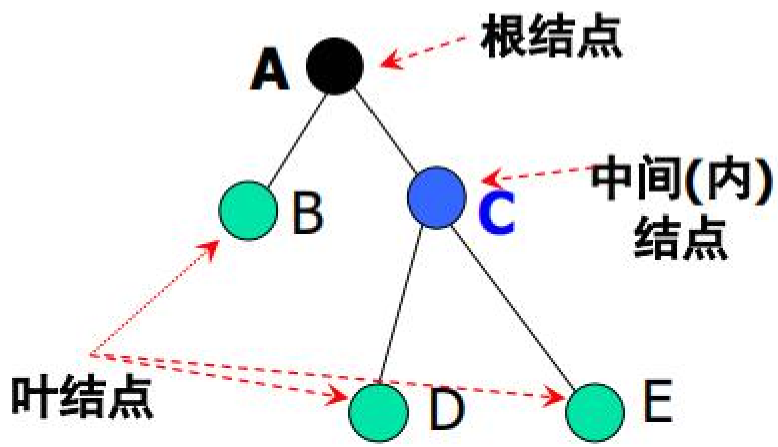
1. 形式语言与自动机基础
2. 隐内马尔模型

1. 形式语言与自动机

1.1 几个基本概念

1.1.1 树 (tree)

一个连通的无回路的无向图称为树(或称自由树)。如果树中有一个结点被特别地标记，则这棵树被称之为根树，这个被特别标记的 结点被称之为根结点。



1.1.2 字符串 (string)

字符串定义：假定Σ是字符的有限集合，它的每一个元素称之为字符。由Σ中字符相连而成的有限序列被称之为Σ上的字符串（或称符号串，或称链）。 特殊地，不包括任何字符的字符串称为空串，记作ε 。

符号串的长度：符号串中符号的个数。符号串x的长度用|x|表示。$|\varepsilon|=0$。包括空串的Σ上字符串的全体记为 Σ^*。

字符串操作：假定 Σ^*是字符的有限集合， x, y 是Σ^*上的符号串

字符串操作：假定Σ 是字符的有限集合， x, y 是Σ 上的符号串

- **字符串连接：**则把y的各个符号写在x的符号之后得到的符号串称为x与y的连接，记作xy。

  例：$\sum\{a,b,c\}, x = ab, y = cba$，则 $xy = abcba$

- 设x是符号串，把x自身连接n次得到的符号串，即 $z = xx\ldots x(n\text{个}x)$，称为x的n次方幂，记作 x^n

  注意：

  - $x^0 = \varepsilon$
  - $x^n = xx^{n-1} = x^{n-1}x(n \geq 1)$
  - $x^* = x^n(n \geq 0)$， $x^+ = x^n(n \geq 1)$

- **符号串集合的乘积**：设A, B是符号串的集合，则A, B的乘积定义为：

$$AB = \{\mathrm{xy} \mid x \in A, y \in B\}$$

相应地，$A^0 = \{\varepsilon\}$ ， $A^{\mathrm{n}} = A^{n-1}A = AA^{n-1}$

- **闭包**：如果V是字符表Σ上的字符串集合，那么，V 的闭包定义为：

$$V^* = V^0 \cup V^1 \ldots$$

注意：

  - $V^+ = V^1 \cup V^2 \ldots$(称为$V$的正闭包)
  - $V^+ = V^* - \{\varepsilon\}$

1.1.3 正则表达式（简称正则式）

正则式对应于Σ上的一些子集（正则集），并通过递归定义：

1）空集φ和空字符串ε是正则式，它们的正则集分别为φ和{ε}。

2）任何 x ∈ Σ, x 是正则式，它对应的正则集是 {x} 。

3）如果X, Y是Σ上的正则式，并且它们对应的正则集分别为U, V，那么，X | Y, X • Y 和 X* 也是正则式，且它们对应的正则集分别为 U ∪ V, U • V 和 U*。

- **正则表达式与有限状态图：**正则表达式可以用有向图表示，图的结点是状态，有一个起始结点和一个终止结点。起始结点只有出边，终止结点用双圆圈表示。边上的符号表示从一个状态到另一个状态结点允许出现的字符，这种图称之为有限状态图。正则式01*

语言描述的三种途径

- 穷举法 —— 只适合句子数目有效的语言。
- 语法描述 —— 生成语言中合格的句子。
- 自动机 —— 对输入的句子进行检验，区别哪些是语言中的句子，哪些不是语言中的句子。

1.2 形式语言

1.2.1 形式语言的直观意义

形式语言是用来精确地描述语言（包括人工语言和自然语言）及其结构的手段。

形式语言学也称代数语言学。

以重写规则 α→β 的形式表示，其中， α，β 均为字符串。顾名思义：字符串 α 可以被改写成 β。一个初步的字符串通过不断地运用重写规则，就可以得到另一个字符串。通过选择不同的规则并以不同的顺序来运用这些规则，就可以得到不同的新字符串。

1.2.2 形式语法的定义

形式语法是一个4元组$G=(N,\Sigma,P,S)$，其中 N 是非终结符的有限集合(有时也叫变量集或句法种类集)；Σ是终结符的有限集合，$N\cap\Sigma=\phi$；$V=N\cup\Sigma$称总词汇表；P 是一组重写规则的有限集合：$P=\{\alpha\rightarrow\beta\}$，其中，α，β 是 V 中元素构成的串，但 α 中至少应含有一个非终结符号；$S\in N$，称为句子符或初始符。

例如： $G=(\{A,S\},\{0,1\},P,S)$

$$P: S\rightarrow 0A1 \qquad 0A\rightarrow 00A1 \qquad A\rightarrow 1$$

1.2.3 形式语法的推导的定义

设 $G=(N,\Sigma,P,S)$ 是一个文法，在（$N\cup\Sigma$）*上定义关系 $\underset{G}{\Rightarrow}$ (直接派生或推导) 如下：

如果 $\alpha\beta\gamma$ 是（$N\cup\Sigma$）*中的符号串，且 $\beta\rightarrow\delta$ 是P的产生式，则 $\alpha\beta\gamma\underset{G}{\Rightarrow}\alpha\delta\gamma$

- **最左推导、最右推导和规范推导**
  - 约定每步推导中只改写最左边的那个非终结符，这种推导称为“最左推导”
  - 约定每步推导中只改写最右边的那个非终结符，这种推导称为“最右推导”
  - 最右推导也称规范推导

例子：设 $G=(\{\mathrm{E},\mathrm{T},\mathrm{F}\},\{\mathrm{a},+,*,(,)\},\mathrm{P},\mathrm{E})$

$$P:\quad E \rightarrow E+T \mid T \quad T \rightarrow T*F \mid F \qquad F \rightarrow (E) \mid \mathrm{a}$$

则，字符串 **a+a*a** 的两种推导过程：

- $E \Rightarrow E+T \Rightarrow T+T \Rightarrow F+T \Rightarrow \mathrm{a}+T \Rightarrow a+T*F$

  $\Rightarrow a+F*F \Rightarrow a+a*F \Rightarrow a+a*a$ (最左推导)

- $E \Rightarrow E+T \Rightarrow E+T*F \Rightarrow E+T*\mathrm{a} \Rightarrow \mathrm{E}+F*a \Rightarrow E+a*a$

  $\Rightarrow T+a*a \Rightarrow F+a*a \Rightarrow a+a*a$ (最右推导)

1.2.4 句型与句子

一些特殊类型的符号串为文法$G=(N,\Sigma,P,S)$的句子形式(句型)：

- S 是一个句子形式
- 如果$\alpha\beta\gamma$ 是一个句子形式，且 $\beta\rightarrow\delta$ 是 P 的产生式，则 $\alpha\beta\gamma$ 也是一个句子形式

文法 G 的不含非终结符的句子形式称为 G 生成的句子。由文法 G 生成的语言，记作 L(G)，指 G 生成的所有句子的集合。即：

$$L(G)=\{\mathrm{x}\mid x\in\Sigma,\mathrm{S}\overset{*}{\underset{\mathrm{G}}{\Rightarrow}}x\}$$

1.2.5 正则文法

如果文法 $G=(N,\Sigma,P,S)$ 的 P 中的规则满足如下形式: $A \rightarrow B\mathrm{x}$，或 $A \rightarrow \mathrm{x}$，其中 $A,B \in N$， $\mathrm{x} \in \Sigma$，则称该文法为正则文法（简写为 FSG）或称3型文法（左线性正则文法）；如果 $A \rightarrow \mathrm{x}B$，则该文法称为右线性正则文法。

1.2.6 上下文无关文法（CFG, context-free grammar)

如果 P 中的规则满足如下形式：$A \rightarrow \alpha$，其中 $A \in N$， $\mathrm{a} \in (N \cup \Sigma)^*$，则称该文法为上下文无关文法（CFG）或称 2 型文法。

- **二义性**: 一个文法 G，如果存在某个句子有不只一棵分析树与之对应，那么称这个文法是二义的。

1.2.7 上下文有关文法（CSG, context-sensitive grammar）

如果 P 中的规则满足如下形式: $\alpha A\beta \to \alpha\gamma\beta$，其中 $A \in N$， $\alpha,\beta,\gamma \in (N \cup \Sigma)^*$，且 γ 至少包含一个字符，则称该文法为上下文有关文法（CSG）或称 1 型文法。

另一种定义： $if x \to y, x \in (N \cup \Sigma)^+, \mathrm{y} \in (N \cup \Sigma)^*, and\ |\mathrm{y}| \geq |x|$

1.2.8 无约束文法（无限制重写系统）

如果 P 中的规则满足如下形式: $\alpha \to \beta$，α， β 是字符串，则称 G 为无约束文法，或称 0型文法。

显然，每一个正则文法都是上下文无关文法，每一个上下无关文法都是上下文有关文法，而每一个上下文有关文法都是 0 型文法。

$$L(G0) \supseteq L(G1) \supseteq L(G2) \supseteq L(G3)$$

1.2.9 CFG 产生的语言句子的派生树表示

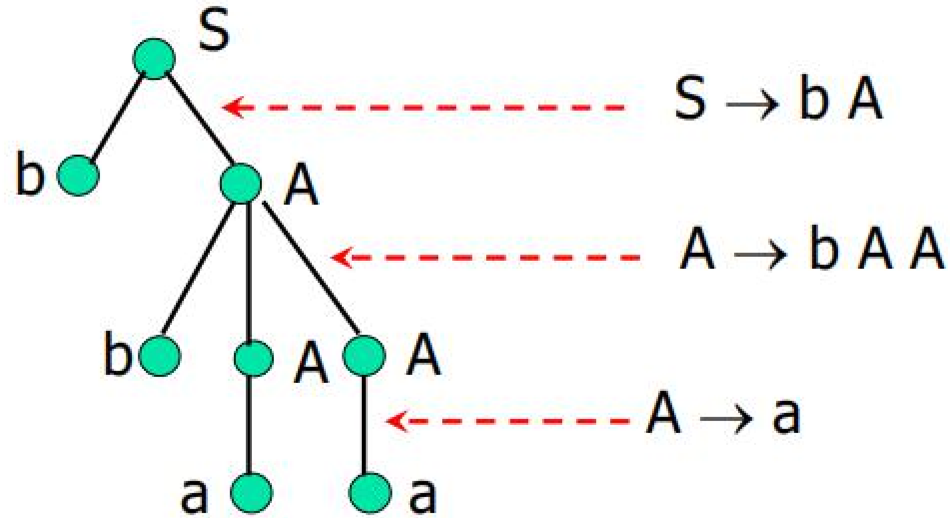
CFG $G=(N,\Sigma,P,S)$ 产生一个句子的派生树由如下步骤构成：

- 对于$\forall \mathrm{x} \in N \cup \Sigma$给一个标记作为节点, S 作为树的根节点
- 如果一个节点的标记为 A，并且它至少有一个除它自身以外的后裔，则 $\mathrm{A} \in N$
- 如果一个节点的标记为 A，它的 k (k > 0) 个直接后裔节点按从左到右的次序依次标记为 $A_1, A_2, \ldots, A_k$ ，则 $A \rightarrow A_1, A_2, \ldots, A_k$ 一定是 P 中的一个产生式

例如，G = ({S, A}, {a, b}, P, S)

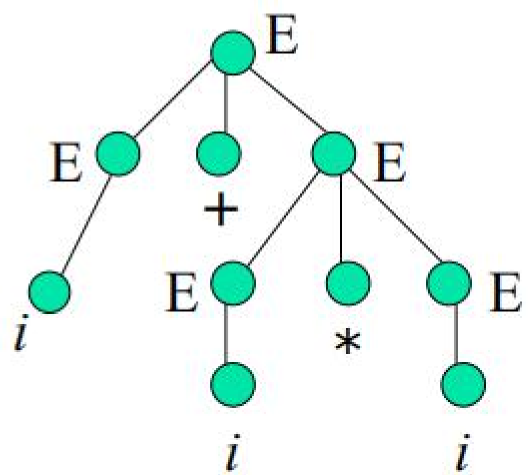
P: $S \to bA \quad A \to bAA \quad A \to a$

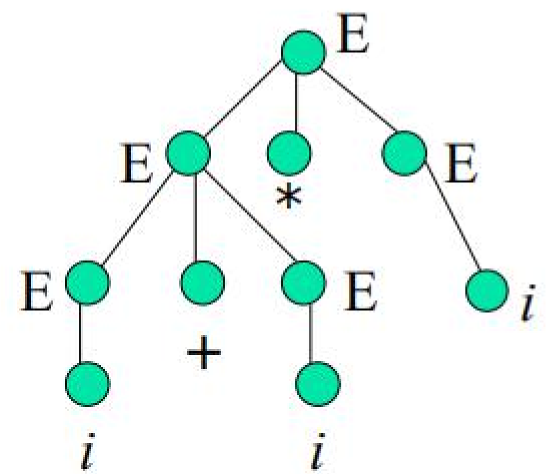
G 所产生的一个句子 bbaa 可以由下面的 生树表示：



例： G(E): $E \rightarrow E + E \mid E * E \mid (E) \mid E - E \mid i$

对于句子 $i+i*i$ 有两棵对应的分析树。





上下文无关文法的二义性

1.3.1 确定的有限自动机 (Definite Automata, DFA)

确定的有限自动机 M 是一个五元组：$M = (\Sigma, Q, \delta, q_0, F)$

- Σ 是输入符号的有穷集合
- Q 是状态的有限集合
- $q_0 \in Q$ 是初始状态
- F 是终止状态集合，$F \subseteq Q$
- δ 是 Q 与 Σ 的直积 $Q \times \Sigma$ 到 Q (下一个状态) 的映射。它支配着有限状态控制的行为，有时也称为状态转移函数

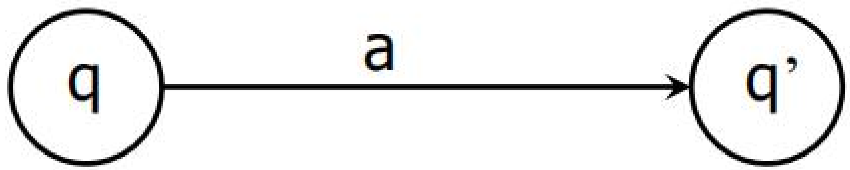
1.3.1 确定的有限自动机 (Definite Automata, DFA)

确定的有限自动机 M 是一个五元组：$M = (\Sigma, \mathrm{Q}, \delta, \mathrm{q}_0, F)$

- Σ 是输入符号的有穷集合
- Q 是状态的有限集合
- $\mathrm{q}_0 \in \mathrm{Q}$ 是初始状态
- F 是终止状态集合，$\mathrm{F} \subseteq \mathrm{Q}$
- δ 是 Q 与 Σ 的直积 $\mathrm{Q} \times \Sigma$ 到 Q (下一个状态) 的映射。它支配着有限状态控制的行为，有时也称为状态转移函数

状态变换图

映射 $\delta(q,a)=q'$ 可以由状态变换图描述



为了明确起见，

终止状态用双圈表示，起始状态用有“开始”标记的箭头表示

DFA 定义的语言

如果一个句子 x 使得有限自动机 M 有

$$\delta(q, \mathrm{x}) = p, p \in F$$

那么，称句子 x 被 M 接受。由 M 定义的语言 T(M) 就是被 M 接受的句子的全集。即

$$\mathrm{T(M)} = \{\mathrm{x} \mid \delta(\mathrm{q_0}, \mathrm{x}) \in \mathrm{F}\}$$

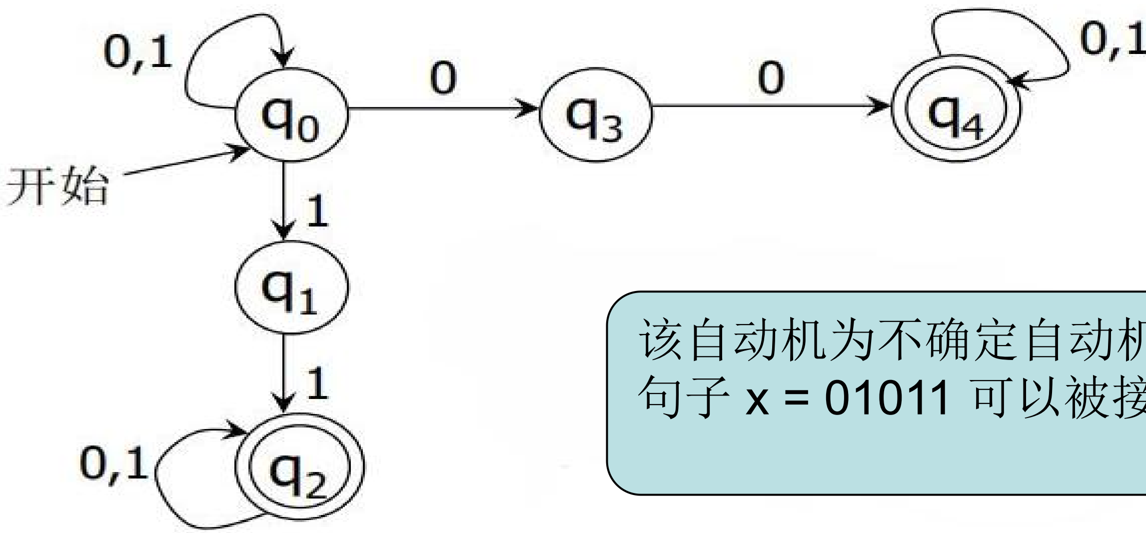
1.3.2 不确定的有限自动机(Non-definite Automata, NFA)

不确定的有限自动机 M 是一个五元组 $M = (\Sigma, Q, \delta, q_0, F)$

- Σ 是输入符号的有穷集合
- Q 是状态的有限集合
- $q_0 \in Q$ 是初始状态
- F 是终止状态集合，$F \subseteq Q$
- δ 是 Q 与 Σ 的直积 $Q \times \Sigma$ 到到Q的幂集 2^Q 的映射

NFA 与 DFA 的区别

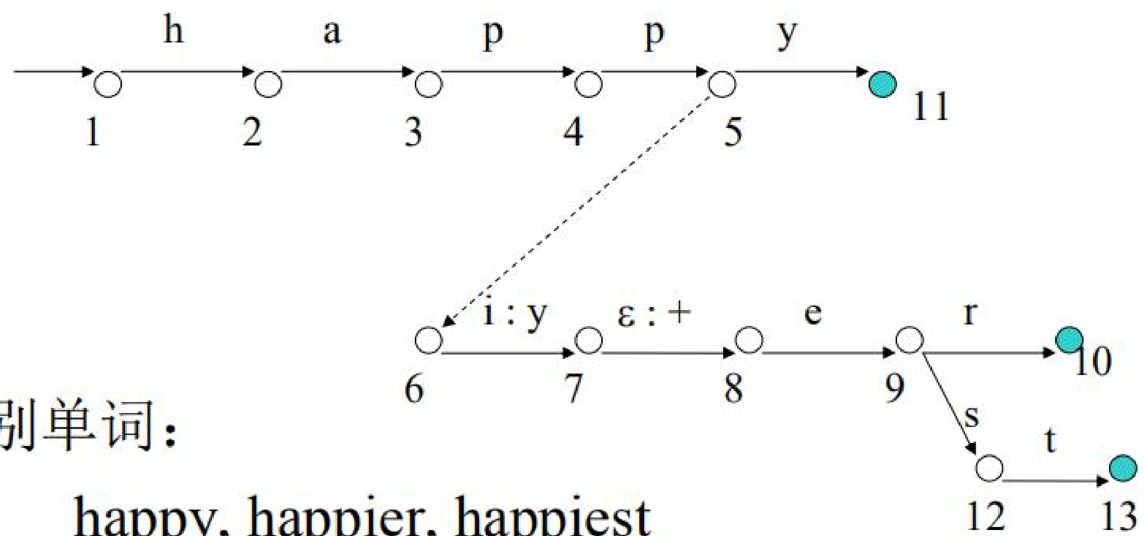
NFA 与 DFA 的唯一区别是：在 NFA 中$\delta(q,a)$是一个状态集合，而在 DFA 中 $\delta(q,a)$ 是一个状态



该自动机为不确定自动机；
句子 x = 01011 可以被接受

有限自动机的应用

一般地，具有相同的前缀或词根，词缀不同的单词可以共用一个有限状态转移机，共享其中的某些状态节点。



识别单词：

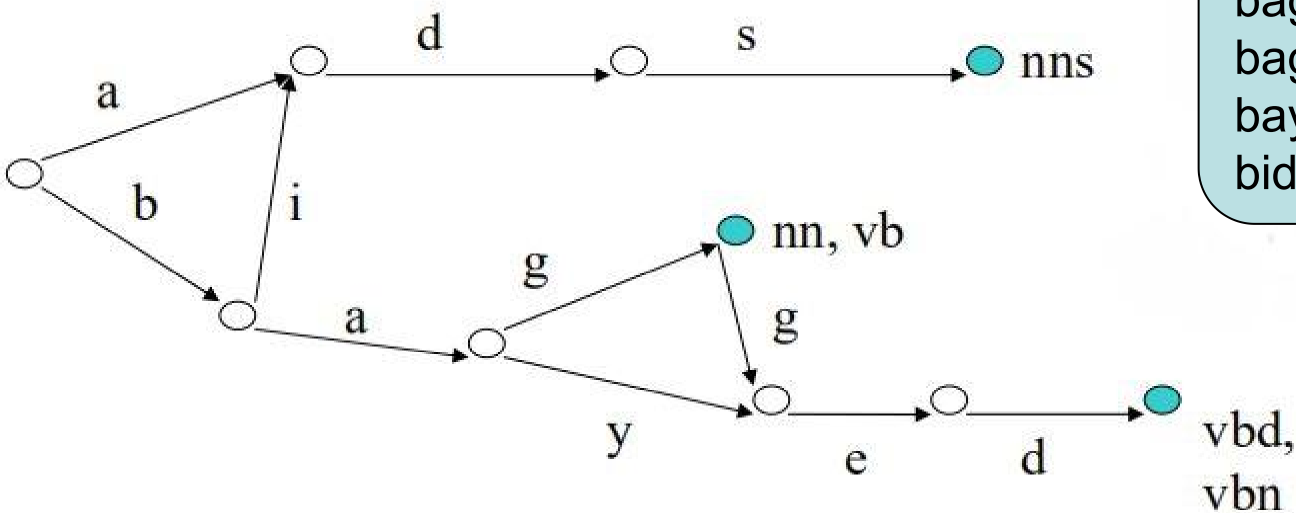
happy, happier, happiest

可转换的形式：

happier → happy + er happiest →happy + est

有限自动机用于词性标注 (lexical tagger)

- DAG: directed acyclic graph

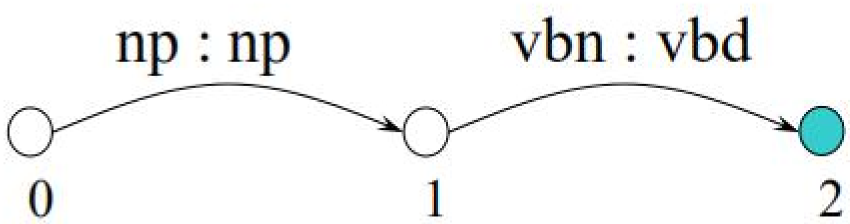


ads	nns
bag	nn, vb
bagged	vbn, vbd
bayed	vbn, vbd
bids	

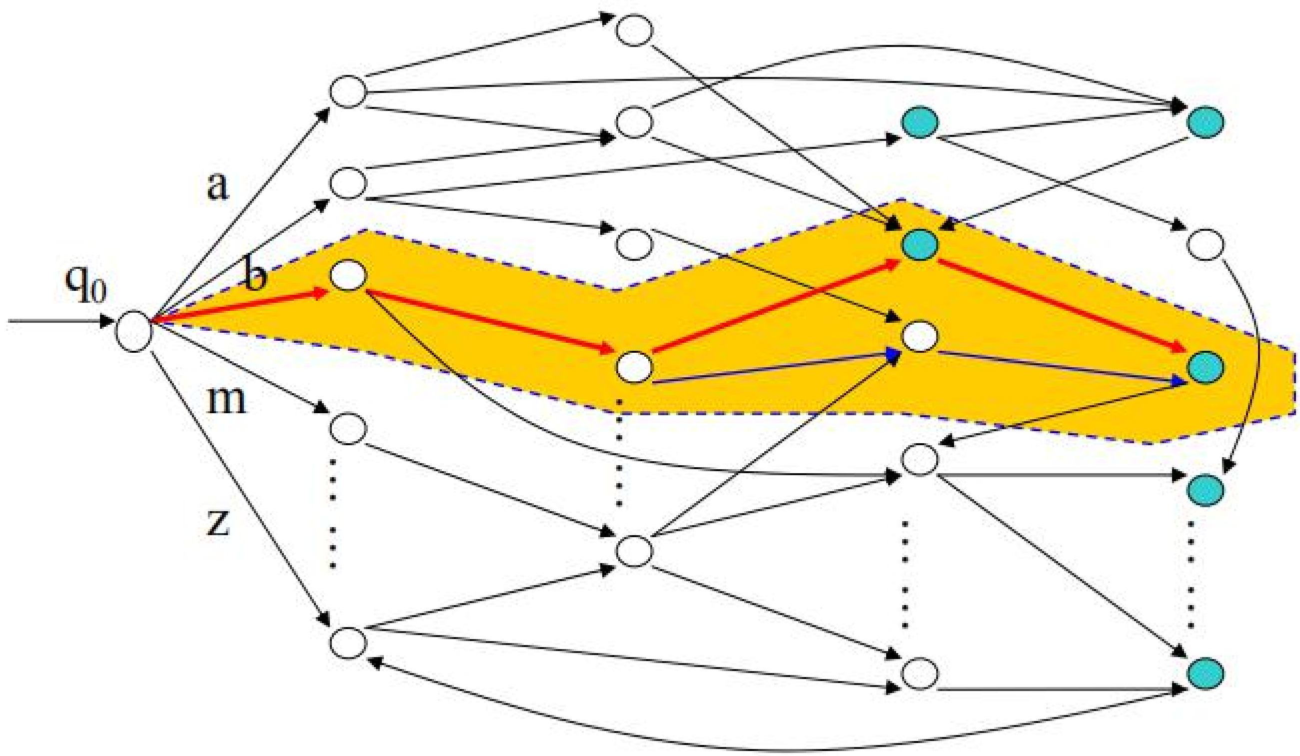
FSTs: 词性上下文约束规则

A B pretag C

如果前一个词性标注是C，那么，将A 转换成 B



对于某一字符串 X，搜索与之编辑距离最短的单词(路径)



正则文法与有限自动机的关系

若$G = (VN, VT, P, S)$是一个正则文法，则存在一个有限自动机 $M = (\Sigma, Q, \delta, q_0, F)$

使得：T(M) = L(G)

对于任意一正则文法，总可以构造一个识别器 ——DFA

语言与识别器的对应关系

识别器是有穷地表示无穷语言的另一种方法。每一个语言的句子都能被一定的识别器所接受。

语言类型	识别器类型
0 型	图灵机
1 型	线性带限自动机
2 型	下推自动机
3 型	有限自动机

2. 马尔可夫模型

马尔可夫模型(Morkov Model)

- 描述

　　存在一类重要的随机过程：如果一个系统有N个状态 $S_1, S_2, \ldots, S_{\mathrm{n}}$ ，随时间推移，该系统从某一状态转移到另一状态。系统在时间 t 的状态记为 q_t 。系统在时间 t 处于状态 $S_j(1 \le j \le N)$的概率取决于其在时间1,2,...,t-1的状态，该概率为：

$$P(\mathrm{q}_t = S_j \mid q_{t-1} = S_i, q_{t-2} = S_k, \ldots)$$

- **假设1：** 如果在特定情况下，系统在时间 t 的状态只与其在时间 t-1 的状态相关，则该系统构成一个离散的一阶马尔柯夫链：

$$P(q_t = S_j \mid q_{t-1} = S_i, q_{t-2} = S_k, \ldots) = P(q_t = S_j \mid q_{t-1} = S_i)$$

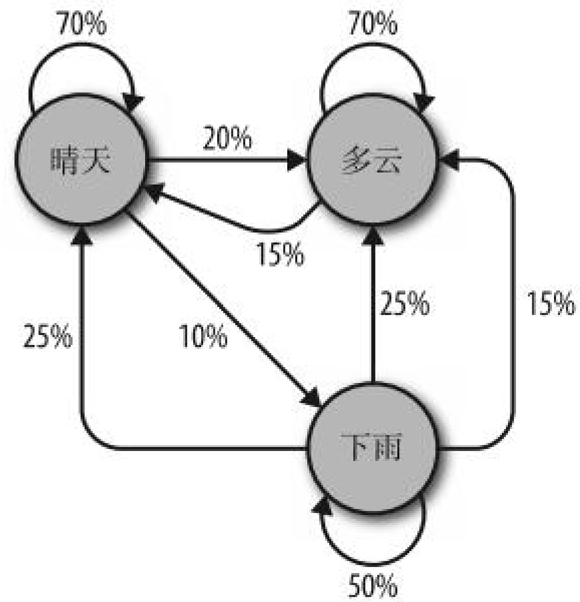
- **假设2：** 如果只考虑独立于时间 t 的随机过程，即所谓的不动性假设，状态与时间无关，那么：

$$P(q_t = S_j \mid q_{t-1} = S_i) = a_{ij} \quad 1 \le i, j \le N$$

该随机过程称为**马尔可夫模型 (Markov Model)**

状态转移概率 a_{ij} 必须满足下列条件：$a_{ij} \ge 0$ 和 $\sum_{j=1}^{N} a_{ij} = 1$

- 马尔可夫模型又可视为**随机有限状态自动机**，该有限状态自动机的每一个状态转换过程都有一个相应的概率，该概率表示自动机采用这一状态转换的
- 马尔可夫链可以表示成状态图(转移弧上有概率的非确定的有限状态自动机)
    - 零概率的转移弧省略
    - 每个节点上所有发出弧的概率之和等于1



马尔柯夫描述的天气模型

状态序列 $S_1,\ldots,S_T$ 的概率：

$$
\begin{aligned}
& P(S_1,\ldots,S_T)=P(S_1)P(S_2\mid S_1)P(S_3\mid S_1,S_2)\ldots P(S_T\mid S_1,\ldots,S_{T-1}) \\
& =P(S_1)P(S_2\mid S_1)P(S_3\mid S_2)\ldots P(S_T\mid S_{T-1}) \\
& =\pi_{S_1}\prod_{t=1}^{T-1}a_{S_tS_{t+1}}
\end{aligned}
$$

其中，$\pi_{\mathrm{i}}=P(q_1=S_i)$ 为初始状态的概率

实例：一段文字中名词、动词、形容词三类词性出现的情况可由三个状态的马尔可夫模型描述。状态s1：名词；状态s2：动词；状态s3：形容词；假设状态之间的转移矩阵如下：

$$
\mathbf{A} = [a_{ij}] = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \end{array}
\begin{array}{c}
\begin{array}{ccc} s_1 & s_2 & s_3 \end{array} \\
\begin{bmatrix} 0.3 & 0.5 & 0.2 \\ 0.5 & 0.3 & 0.2 \\ 0.4 & 0.2 & 0.4 \end{bmatrix}
\end{array}
$$

如果在该段文字中某个句子的第一个词为名词，那么根据这一模型M，在该句子中这三类词出现顺序为O=”名动形名”的概率为：

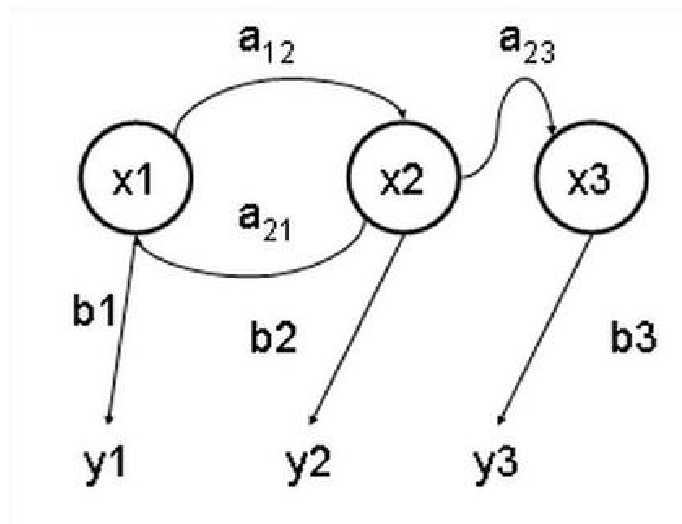
$$
\begin{aligned}
&P(O \mid M) = P(\mathrm{s}_1, s_2, s_3, s_1 \mid M) \\
&= P(s_1) \times P(s_2 \mid s_1) \times P(s_3 \mid s_2) \times P(s_1 \mid s_3) \\
&= 1 \times a_{12} \times \mathrm{a}_{23} \times a_{31} \\
&= 0.5 \times 0.2 \times 0.4 \\
&= 0.04
\end{aligned}
$$

隐马尔可夫模型(Hidden Markov Model, HMM)

描述：在隐马尔柯夫模型中，我们不知道模型所经过的状态序列，只知道状态的概率函数，也就是说，观察到的事件是状态的随机函数；因此，该模型是一个双重的随机过程。其中，模型的状态转换是不可观察的，即隐蔽的，可观察事件的随机过程是隐蔽的状态转换过程的随机函数

隐马尔可夫模型状态的变迁可以表示为:

- x —— 隐含状态
- y —— 可观察的输出
- a —— 转换概率（transition probability）
- b —— 输出概率（emission probability）



隐马尔可夫模型状态变迁图（例子）

HMM形式定义

设Q($Q = q_1, q_2, ..., q_N$)是所有可能的状态的集合，V($V = v_1, v_2, ..., v_M$)是所有可能的观测的集合；其中N是可能的状态数，M是可能的观测数

设I($I = i_1, i_2, ..., i_T$) 是长度为T的状态序列，O($O = o_1, o_2, ..., o_T$)是对应的观测序列

- A是状态转移矩阵（一个时刻一个状态转移矩阵）：$A = [a_{ij}]_{N \times N}$
  - 在时刻t，处于 q_i 状态的条件下，时刻t+1转移到状态 q_j 的概率为：

$$a_{ij} = P(i_{t+1} = q_j \mid i_t = q_i)$$

- B是观测概率矩阵：$B = [b_j(k)]_{N \times M}$
  - 在时刻t处于状态 q_j 的条件下生成观测 v_k 的概率为：

$$b_j(k) = P(o_t = v_k \mid i_t = q_j)$$

- 初始状态的概率分布向量为：$\pi = (\pi_i)$ ，其中 $\pi_i = P(i_1 = q_i)$

隐马尔科夫模型由初始状态概率向量π、状态转移概率矩阵A和观测概率矩阵B决定。π和A决定状态序列，B决定观测序列。因此，隐马尔科夫模型可以由三元符号表示，即

$$\mu = (A, B, \pi)$$

HMM 中的三个问题

- 在给定模型 $\mu=(A,B,\pi)$和观察序列$O=\mathrm{o}_1,\mathrm{o}_2,\ldots,\mathrm{o}_T$情况下，怎样快速计算概率 $P(O\mid\mu)$？
- 在给定模型$\mu=(A,B,\pi)$和观察序列$O=\mathrm{o}_1,\mathrm{o}_2,\ldots,\mathrm{o}_T$情况下，如何选择在一定意义下“最优”的状态序列$Q=\mathrm{q}_1,\mathrm{q}_2,\ldots,\mathrm{q}_N$，使得该状态序列 “最好地解释”观察序列?
- 给定一个观察序列$O=\mathrm{o}_1,\mathrm{o}_2,\ldots,\mathrm{o}_T$，如何根据最大似然估计来求模型的参数值？即如何调节模型$\mu=(A,B,\pi)$的参数，使得 $P(O\mid\mu)$最大?

隐马尔柯夫模型（拓展）：

- 3个问题:
  - ✓ 快速计算给定模型的观察序列的概率
    - 向前算法或向后算法
  - ✓ 求最优状态序列
    - Viterbi 算法
  - ✓ HMM 中的参数估计
    - Baum-Welch (向前向后)算法
- 模型实现中需要注意的问题：小数溢出

1. 形式语言与自动机

- 几个基本概念
  - 树，字符串，字符串操作
  - 正则表达式，有限状态图
- 自动机
  - 有限自动机
  - 应用

2. 马尔可夫模型

  - 马尔可夫模型
  - 隐马尔可夫模型

Thank you！