词法分析：

1. 不确定的有限状态自动机 NFA

① 有穷状态集合 S

② 输入符号集合 Σ （且 $\varepsilon \notin \Sigma$）

③ 转换函数 $move: S \times (\Sigma \cup \{\varepsilon\}) \to P(S)$

④ 唯一的开始状态 $s_0 \in S$

⑤ 接受状态集合 $F \subseteq S$

2. 确定的有限状态自动机 DFA

① 有限状态集合 S

② 输入符号集合 Σ （且 $\varepsilon \notin \Sigma$）

③ 转换函数 $move: S \times \Sigma \to S$ 且可以是部分函数

④ 唯一的开始状态 $s_0 \in S$

⑤ 接受状态集合 $F \subseteq S$

3. NFA 与 DFA 的差别

① DFA 没有输入 ε 上的转换

② $\forall$ 状态 s 和输入符号 a，在 DFA 中有且只有一条标号为 a 的边离开 s

4. 从 NFA → DFA 子集构造法

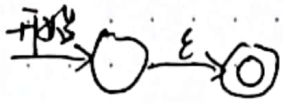
NFA 开始状态求 ε 闭包 ⇒ DFA 开始状态

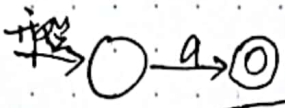
下一个状态集的 ε 闭包 ↓

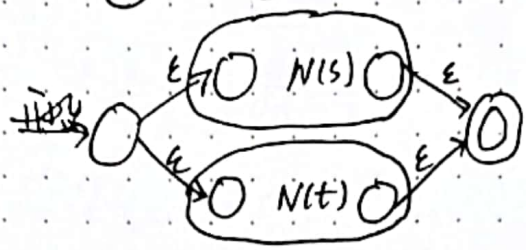
下一个 DFA 状态

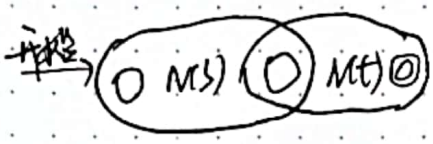
包含 NFA 的终结状态的 DFA 状态也终结状态

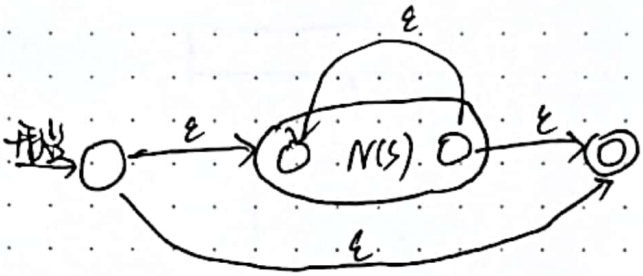
5. 正则式 → NFA. (MYT分法).

$r=\varepsilon, r=(s)$ 

$r=a$ 

$r=s|t$ 

$r=st$ 

$r=s^*$ 

6. 正则式 → DFA.

① 扩展正则式 r#，构建语法树.

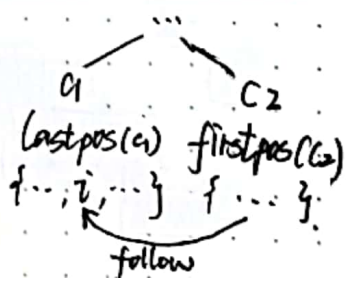
② (firstpos 和 lastpos 大体相同)

	nullable.	firstpos. lastpos
ε	true	Φ
i	false	{i}
$n=c_1\|c_2$	or.	firstpos(c_1) ∪ firstpos(c_2).
$n=c_1c_2$	and.	if (nullable(c_1)) firstpos(c_1) ∪ firstpos(c_2). else firstpos(c_1). [lastpos c_1和c_2对调]
$n=c^*$	true	firstpos(c_1).

③ followpos: ∀i ∈ lastpos(c_1).

firstpos(c_2) ⊆ followpos(i)

条件 $n=c_1c_2$ 或 $n=c_1^*$ (此时 $c_1=c_2$).



④构造算法：初始状态：firstpos(n_0)：n_0为语法树根结点.

后续状态：和输入符号 a 对应的所有位置 p.

$\bigcup_p follow(p)$.

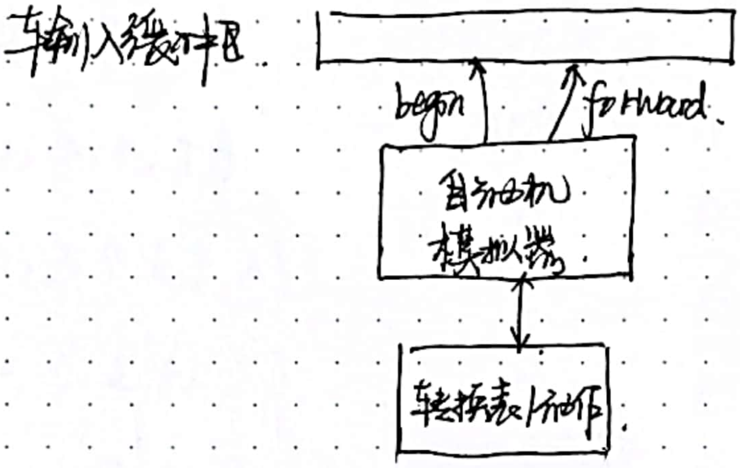
7. DFA最简化：

{非终结状态}　{终结状态}

依次检查各个输入符号下的转换是否仍在这一组中.

若不在，则细分．若全在，则结束．

8. 词法分析器的模型.



语法分析：

1. 上下文无关文法 CFG 是一个四元组 (V_T, V_S, S, P)

(1) V_T：终结符集合

(2) V_S：非终结符集合

(3) S：开始符号 $S \in V_S$

(4) P：产生式集合

2. 消除左递归：$A \to A\alpha | \beta$

$\Rightarrow$ $A \to \beta A'$

$A' \to \alpha A' | \varepsilon$

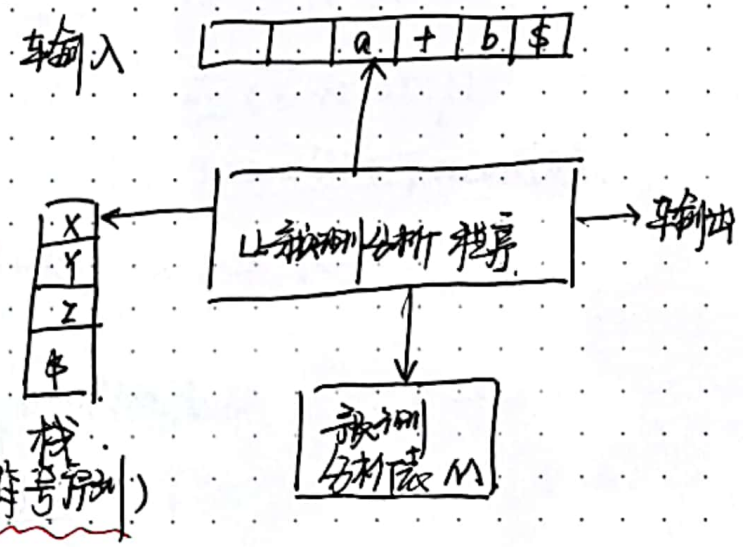
3. 提取左公因子：$A \to \alpha\beta_1 | \alpha\beta_2$

$\Rightarrow$ $A \to \alpha A'$

$A' \to \beta_1 | \beta_2$

消除二义性

4. 下推自动机语法分析模型（LL语法分析）



注：判断文法是否有二义性：是当有多种语法分析树对应一个符号串

5. LL(1)文法：

① 向前看一个输入符号

② 无左公因子

③ 不是二义性的

④ 不含左递归。

6. 预测分析表构建 (LL(1)). ☆ 计算顺序：从下到上.

(1) FIRST(A) $A = X_1 X_2 \cdots X_n$.

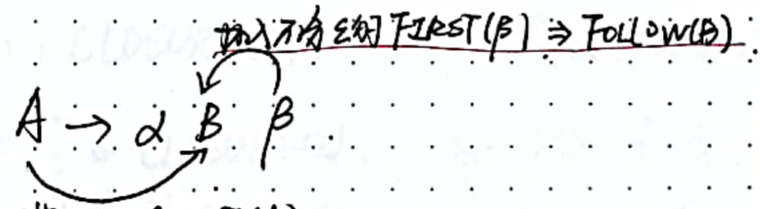
加入 $FIRST(X_1)$ 中所有非ε符号

若 $\varepsilon \in FIRST(X_1)$，加入 $FIRST(X_2)$ 中所有非ε符号

以此类推。若 X_i 为终结符，则 $FIRST(X_i) = \{X_i\}$.

若 $\forall X_i$, $\varepsilon \in FIRST(X_i)$，则 $\varepsilon \in FIRST(A)$.

(2) FOLLOW： $A = \alpha B \beta$.



若 $\varepsilon \in FIRST(\beta)$.

$FOLLOW(A) \subseteq FOLLOW(B)$.

计算顺序：从上到下.

偷懒法：FIRST(α)中所有都用这个产生式
若FIRST(α)有空串，则FOLLOW(A)中所有也用这个产生式

(3) 构建预测分析表M： $\forall A \to \alpha$.

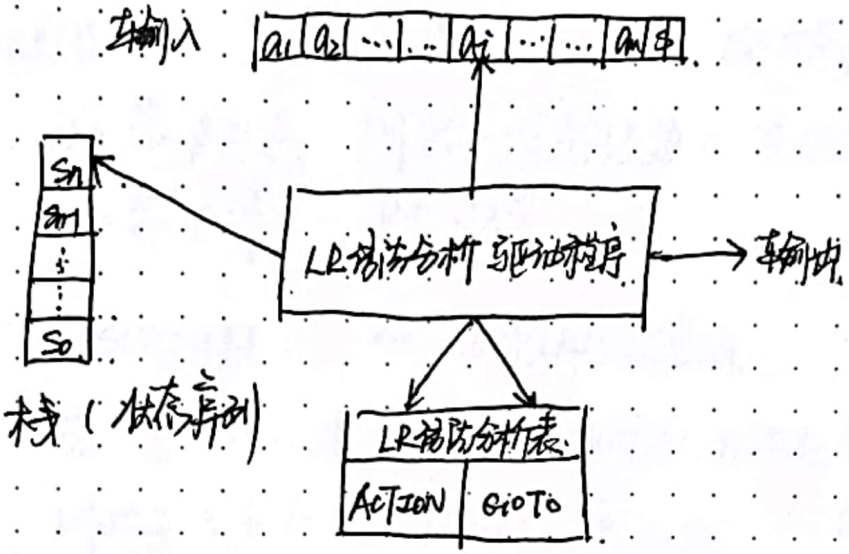
$\forall a \in FIRST(\alpha)$. $M[A, a] = A \to \alpha$.

若 $\varepsilon \in FIRST(\alpha)$. $\forall b \in FOLLOW(A)$. $M[A, b] = A \to \alpha$.
(b包括$).

其余留空，表示出错

若从中出现多重定义，则文法不是LL(1)的（有二义性）.

7. LR语法分析器模型



8. SLR(1)语法分析

① 增广文法：$G = G + \{S' \to S\}$，S为开始符号
当使用 $S' \to S$ 归约时为接受，停止语法分析

② LR(0)项：右部某处加点的产生式。（$A \to \alpha \cdot$）

③ 项集的闭包：CLOSURE(I) ⇒ 构造LR(0)项集族

若 $[A \to \alpha \cdot B\beta] \in$ CLOSURE(I)，$B \to \gamma$ 为一产生式
且 $[B \to \cdot\gamma] \notin$ CLOSURE(I)，则将 $B \to \cdot\gamma$ 加入

④ GOTO表：若 $I_i \xrightarrow{A} I_j$（只针对非终结符）
则 $GOTO[i, A] = j$

计算 FOLLOW

⑤ ACTION表：S_i 表示移进，i为压栈的状态编号
r_i 表示归约，i为使用的产生式编号

若：$[A \to \alpha \cdot a\beta] \in I_i$ 且 $GOTO[i, a] = j$，则 $ACTION[i, a] = S_j$（a为终结符）
若 $[A \to \alpha \cdot] \in I_i$，则 $\forall a \in FOLLOW(A)$，$ACTION[i, a] =$ 归约为 $A \to \alpha$
若 $[S' \to S \cdot] \in I_i$，则 $ACTION[i, \$] = acc$（接受）
若为在冲突则不满足SLR(1)的，其余留空、表示出错

9. 构造 LR(1) 分析表

① LR(1)项： $[A \to \alpha \cdot \beta, a]$，a为向前搜索符号。

<1> 若 $\beta = \varepsilon$，则当下一个输入为 a 才可以归约 $A \to \alpha$。

<2> 若 $\beta \neq \varepsilon$，状态转进。

② CLOSURE(I) ⇒ 构造LR(1)项集族

若 $[A \to \alpha \cdot B\beta, a] \in CLOSURE(I)$，且存在产生式 $B \to \gamma$。

则对于 $\forall b \in FIRST(\beta a)$，若 $[B \to \cdot\gamma, b] \notin CLOSURE(I)$，

则将其加入。

③ GOTO的构建同SLR(1)。

④ ACTION表：

若：$[A \to \alpha \cdot a\beta, b] \in I_i$，且 $GOTO[i, a] = j$

则：$ACTION[i, a] = s_j$。（a为终结符）

若：$[A \to \alpha \cdot, a] \in I_i$，

若 $A = S'$，则 $ACTION[i, \$] = acc$，此时 $A \to \alpha \cdot$ 为 $S' \to S \cdot$

否则 $ACTION[i, a] =$ 归约 $A \to \alpha$。

若存在冲突，则文法不是 构造LR(1)的。

其余留空，表示出错。

I. 判断文法是否为 LL(1) 文法：（充要条件）

文法 G 中任意两个具有相同左部的产生式 $A \to \alpha | \beta$.

① α 和 β 中至多只有一个能推导出 ε.

② 若均不能推导出 ε，则 $FIRST(\alpha) \cap FIRST(\beta) = \phi$.

③ 若 γ 为 α(或 β) 则 $FIRST(\beta(或\alpha)) \cap FOLLOW(A) = \phi$.

即：$\varepsilon \in FIRST(\alpha(或\beta))$

II. 判断文法是否为 LR(0) / SLR(1) / 规范 LR(1) 文法：

构建相应语法分析表，ACTION 表中无冲突项目（移进-归约 / 归约-归约）.

III：LR(0) / SLR(1) / 规范 LR(1) 语法分析表构建方法对比：

相同点：

1. GOTO 表的构建方法相同：GOTO 表只针对非终结符

若 $I_i \xrightarrow{A} I_j$ 则 $GOTO[i, A] = j$.（A 为非终结符）

2. ACTION 表中移进动作相同：

若 $A \to \alpha \cdot a \beta$ ① $\in I_i$ 且 $GOTO[i, a] = j$ ②

则 $ACTION[i, a] = S_j$.

①：LR(0)、SLR(1) 为 LR(0) 项；规范 LR(1) 为 LR(1) 项

②：理解：遇到终结符，且当前项集有产生式接受这个终结符，此时移进

3. 语法分析表的结构相同：ACTION + GOTO.

4. ACTION 表接受动作相同：都是在归约 $S' \to S \cdot$ 时为 acc.

不同点：

1. CLOSURE 计算方法不同.

LR(0) 项：$[A \to \alpha \cdot B \beta]$ 在闭包中，将 $[B \to \cdot \gamma]$ 加入.

LR(1) 项：$[A \to \alpha \cdot B \beta, a]$ 在闭包中，将 $[B \to \cdot \gamma, b]$ 加入.

其中 $b \in FIRST(\beta a)$

注：两者都需要保证存在产生式 $B \to \gamma$

2、ACTION表归约动作不同：

$A \to \alpha$ 为文法的第 j 个产生式：

不作要求 LR(0)：若 $[A \to \alpha \cdot] \in I_i$，则对所有终结符 a（从左至右）

$ACTION[i, a] = r_j$

SLR(1)：若 $[A \to \alpha \cdot] \in I_i$，则 $\forall a \in FOLLOW(A)$，

$ACTION[i, a] = r_j$

规范LR(1)：若 $[A \to \alpha \cdot, a] \in I_i$，则 $ACTION[i, a] = r_j$

总结：从LR(0) - SLR(1) - 规范LR(1)：归约动作的条件越来越严格

LR(0) 将要其个终结符就可以归约

SLR(1) 需要这个终结符 FOLLOW(A)

规范LR(1) 要求向前有符号为 a 才能归约