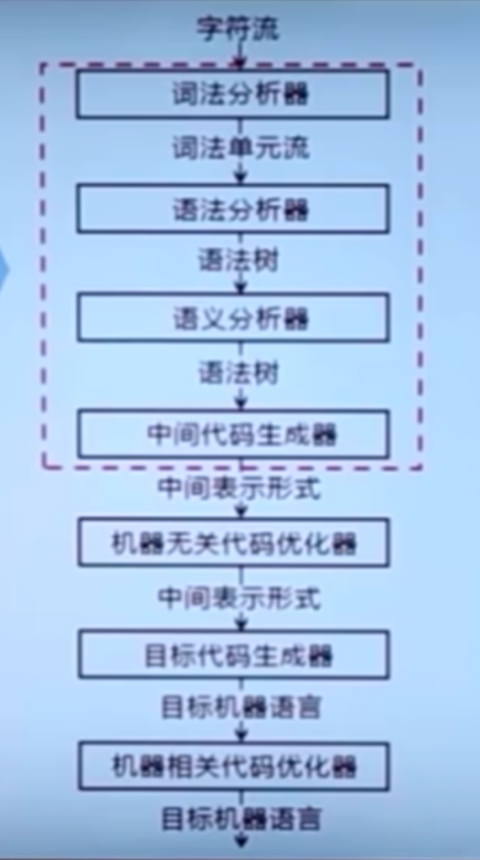
第一章

编译：将高级语言（源语言）翻译成汇编语言或机器语言（目标语言）的过程

分析源语言是先词法分析，再语法分析，最后语义分析

汇编程序：把汇编语言源程序翻译为机器语言程序

编译器结构：



第二章

文法形式化定义

G=（VT,VN,P,S）

VT:终结符集合，文法定义语言的基本符号，也称token

VN：非终结符集合，表示语法成分的符号，也称语法变量

P:产生式集合，一般形式为α->β，α称为头或左部，且至少包含VN中的一个元素，β称为体或右部

S:开始符号，S∈VN，该文法中最大的语法成分

推导：用产生式的右部替换产生式的左部

最左推导：总是选择每个句型的最左非终结符进行替换

最右推导：也叫做规范推导，总是选择每个句型的最右非终结符进行替换

归约：推导的逆过程

最左归约：也叫做规范归约，最右推导的逆过程为最左归约

最右归约：最左推导的逆过程称为最右归约

符号约定：

终结符：

1. 字母表中排在前面的小写字母，如a,b,c
2. 运算符
3. 标点符号
4. 数字
5. 粗体字符串

非终结符：

1. 字母表中排在前面的大写字母，如A,B,C
2. 字母S通常表示开始符号
3. 小写、斜体的名字
4. 代表程序构造的大写字母，如E（表达式）、T（项）、F（因子）

字母表中排在后面的大写字母表示文法符号（终结符或非终结符），排在后面的小写字母是终结符号串（含空串）

小写希腊字母表示文法符号串

除非特别声明，第一个产生式的左部就是开始符号

由文法G开始符号S推导出的所有句子构成的集合称为文法G生成的语言，记为L（G），即

L(G)={w|S⇒\*w,w∈VT\*}

句型，句子：

如果 S⇒\* α，α∈(VT∪VN)\*，则称α是G的一个句型 (sentential form)，一个句型中既可以包含终结符，又可以包含非终结符，也可能是空串。

如果 S⇒\* w，w ∈VT\*，则称w是G的一个句子(sentence)，句子是不包含非终结符的句型。

四种文法：

1. 0型文法（无限制文法）：α中至少包含一个非终结符
2. 1型文法（CSG）（上下文有关文法）：|α|≤|β|
3. 2型文法（CFG）（上下文无关文法）：α∈VN
4. 3型文法（RG）（正则文法）：A→wB或A→w(A→ Bw或A→w)

直接短语一定是某产生是的右部，但产生是的右部不一定是给定句型的直接短语

第三章

正则表达式（RE）：

运算优先级：\*、连接、|（或运算）

对任何正则文法G，存在定义同一语言的正则表达式r

对任何正则表达式r，存在生成同一语言的正则文法G

正则定义：

d1→r1,d2→r2

每个d代表着新符号，且不在字母表Σ中

每个r∈Σ∪{d1,...,d(n-1)}

有穷自动机（FA）：

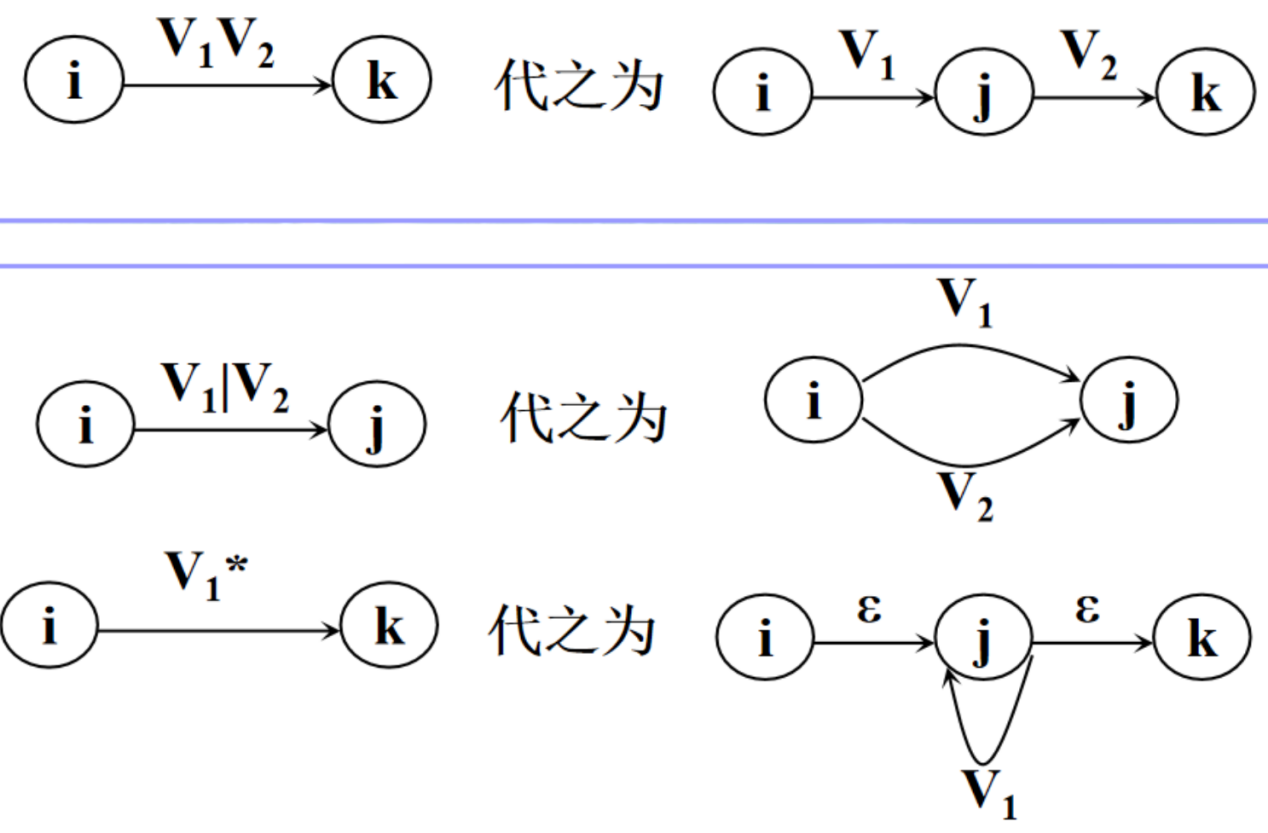
当输入串的多个前缀与一个或多个模式匹配时，总是选择最长的前缀进行匹配

在到达某个终态时，只要输入带上还有符号，FA就继续前进，以便寻找长的匹配

DFA与NFA的区别：

NFA的状态转换过程中可以有空串，所以在给出字符前，无法确定自动机所处的状态，才为不确定的有穷自动机

正则表达式对DFA的转换原则：



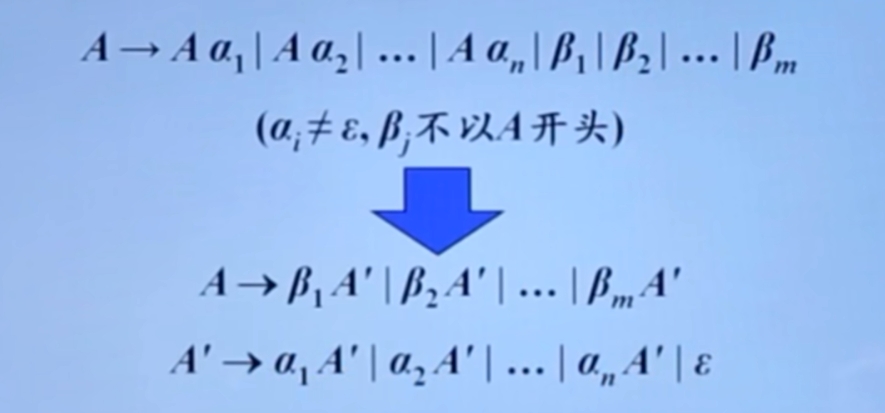
第四章

自顶向下分析：

每一步推导需要做两个选择，第一，替换当前句型的哪个非终结符，第二，用该终结符的哪个候选式进行替换

最左（最右）推导：总是选择每个句型的最左（最右）非终结符进行替换

消除直接最左递归的一般形式：



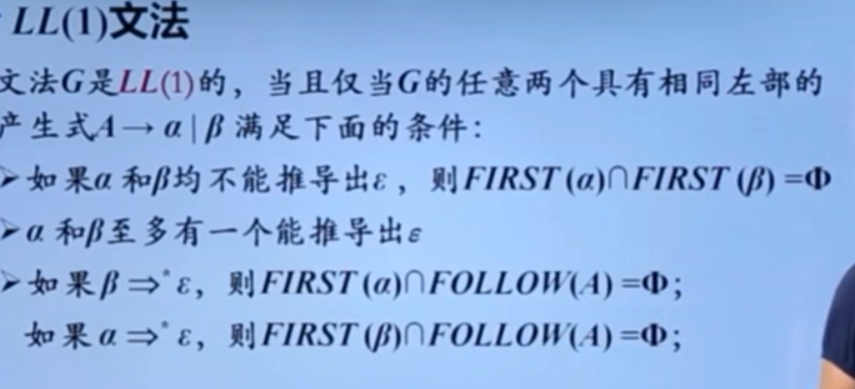
LL(1)文法：

q\_文法：

1. 每个产生式右部为空串或以终结符开始 2）具有相同左部的产生式有不相交的可选集

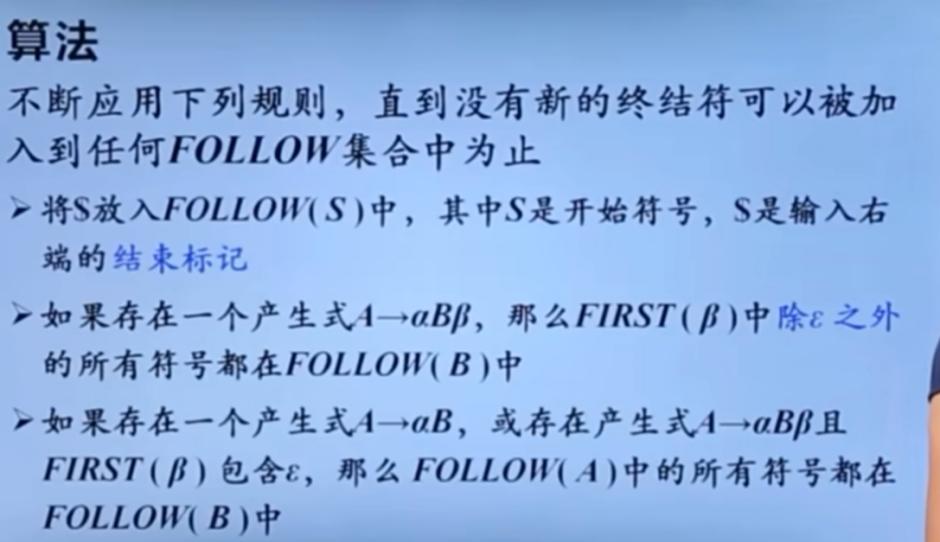
s\_文法：

每个产生是的右部都以终结符开始，同一非终结符的各个候选式的首终结符都不相同



Follow集：

若A终结符是某个句型的的最右符号，则follow(A)依赖于该产生式的左部follow集



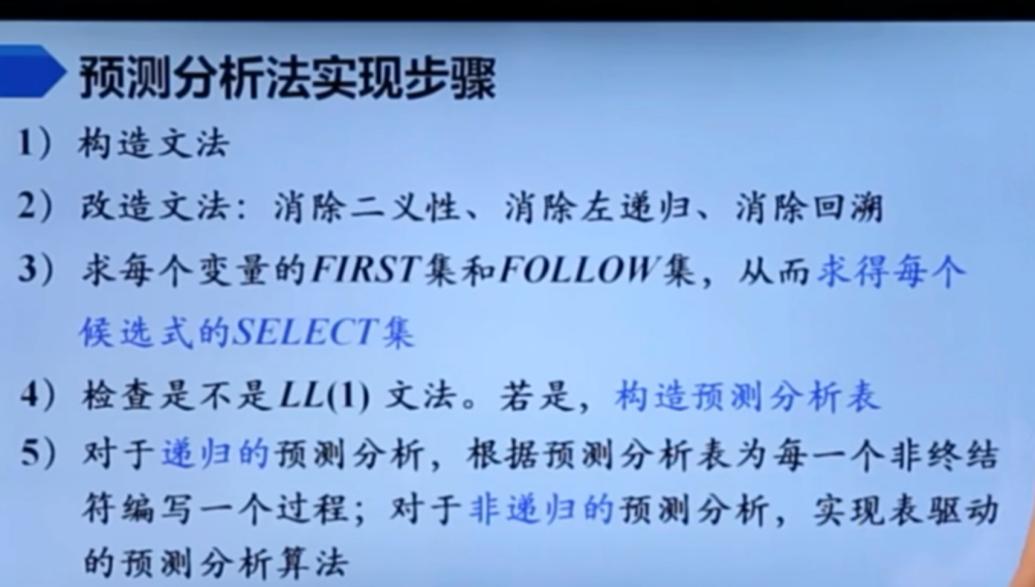
1.先写产生式右部第一个非终结符的follow集

2.将产生式左部follow集加入到产生式右部最后一位非终结符follow集中（产生式右部最后一位是非终结符）

3.若产生式右部最后一位可为空串，将产生式左部follow集加入到产生式右部倒数第二非终结符follow集

4.Follow集中无空串

预测分析法实现步骤：



递归预测分析：直观性强

非递归预测分析：程序规模小、效率更高、难易程度较易

自底向下的语法分析：可以看成将输入串w规约为文法开始符号S的过程，且采用最左规约的方式，其通用框架为移入-规约分析

移入规约分析例子：

