编译器的构造

- 前端
  - **词法分析**：输入源程序，输出记号流
    将程序字符流分解为记号（Token）流，<token_name, attribute>
    非形式描述的语言→正则表达式→NFA→DFA→化简DFA
  - **语法分析**：输入记号流，输出语法树
    也称为解析，在词法记号的基础上，创建语法结构。需要区分合法与非法的记号序列
  - **语义分析**：输入语法树，输出带注解的语法树
    编译器会检查程序中的不一致
  - **中间代码生成器**：输入带注解的语法树，输出中间表示
    是源语言与目标语言之间的桥梁
- 后端
  - **代码优化器**
  - **代码生成器**：输入中间表示，输出目标程序

词法分析

1. 串和语言

- 术语
  - 字母表： 符号的有限集合，例： ∑ = { 0, 1}
  - 串： 符号的有穷序列，例： 0110, ε
  - 语言： 字母表上的一个串集，例：{ε, 0, 00, 000, ...}, {ε}, ∅
  - 句子： 属于语言的串
- 串的运算 **优先级：闭包*〉 连接〉 选择 |**
  - 连接（积）： xy， sε = εs = s
  - 指数（幂）： s^0为ε， s^i为$s^{i-1}s$（i > 0）
- 语言的运算 **优先级：幂〉 连接〉 并**
  - 并： L ∪ M = {s | s ∈L 或 s ∈ M }
  - 连接： LM = {st | s ∈ L 且 t ∈ M}
  - 幂： L^0是{ε}， L^i是$L^{i-1}L$
  - 闭包： $L^*> = L^0 \cup L^1 \cup L^2 \cup ...$
  - 正闭包： $L^+ = L^1 \cup L^2 \cup ...$

2. 有限自动机

- 不确定的有限自动机（简称NFA）包括
  - 有限的状态集合S
  - 输入符号集合∑
  - 转换函数move : **S × (∑∪{ε}) → P(S)**
  - 状态s_0是唯一的开始状态
  - F ⊆ S是接受状态集合
- 确定的有限自动机（简称DFA）包括
  - 有限的状态集合S
  - 输入符号集合∑

- 转换函数**move : S × ∑ → S**， 且可以是部分函数
- 状态s_0是唯一的开始状态
- F ⊆ S是接受状态集合

- NFA与DFA的区别
  - NFA允许空串输入
  - NFA跳转到的状态是状态的幂集，可以有多个选择

语法分析

1. 上下文无关文法CFG

- CFG优点与缺点
  - 优点
    - 文法给出了精确的，易于理解的语法说明
    - 自动产生高效的分析器
    - 可以给语言定义出层次结构
    - 以文法为基础的语言的实现便于语言的修改
  - 缺点
    - 文法只能描述编程语言的大部分语法

❑为什么要用正则表达式定义词法

❖词法规则非常简单，不必用上下文无关文法。

❖对于词法记号，正则表达式描述简洁且易于理解。

❖从正则表达式构造出的词法分析器效率高。

❑分离词法分析和语法分析的好处（软件工程视角）

❖简化设计

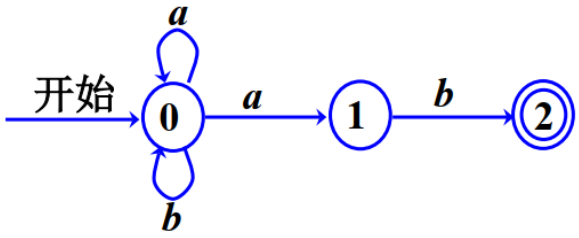
❖编译器的效率会改进

❖编译器的可移植性加强

❖便于编译器前端的模块划分

- NFA →上下文无关文法
  - 确定终结符集合
  - 为每个状态引入一个非终结符Ai
  - 如果状态i有一个a转换到状态j,引入产生式Ai→aAj,如果i是接受状态，则引入Ai→ε

$$A_0 \to a\,A_0 \mid b\,A_0 \mid a\,A_1$$
$$A_1 \to b\,A_2$$
$$A_2 \to \varepsilon$$



2. 推导

- 最左推导（leftmost derivation）：每步代换最左边的非终结符

$$E \Rightarrow_{lm} -E \Rightarrow_{lm} -(E) \Rightarrow_{lm} -(E+E)$$
$$\Rightarrow_{lm} -(\mathrm{id}+E) \Rightarrow_{lm} -(\mathrm{id}+\mathrm{id})$$

- 最右推导（rightmost or canonical derivation）：每步代换最右边的非终结符

$$E \Rightarrow_{rm} -E \Rightarrow_{rm} -(E) \Rightarrow_{rm} -(E+E)$$
$$\Rightarrow_{rm} -(E+\mathrm{id}) \Rightarrow_{rm} -(\mathrm{id}+\mathrm{id})$$

3. 语言、文法、句型、句子

- 上下文无关文法G产生的语言：从开始符号S 出发，经 ⇒+推导所能到达的**所有仅由终结符组成**的串
- 句型： S ⇒*α， S是开始符号，α 是**由终结符和/或非终结符组成**的串， 则α是文法G的句型
- 句子： **仅由终结符组成**的句型

4. 文法的二义性

- 文法的某些句子存在不止一种最左(最右)推导，或者不止一棵分析树，则该文法是二义的。
- 消除二义性：可通过定义**运算优先级和结合律**来消除二义性

❑新的非二义文法

$$E \rightarrow E+T \mid T$$
$$T \rightarrow T*F \mid F$$
$$F \rightarrow \mathrm{id} \mid (E)$$