

Lecture 2.1

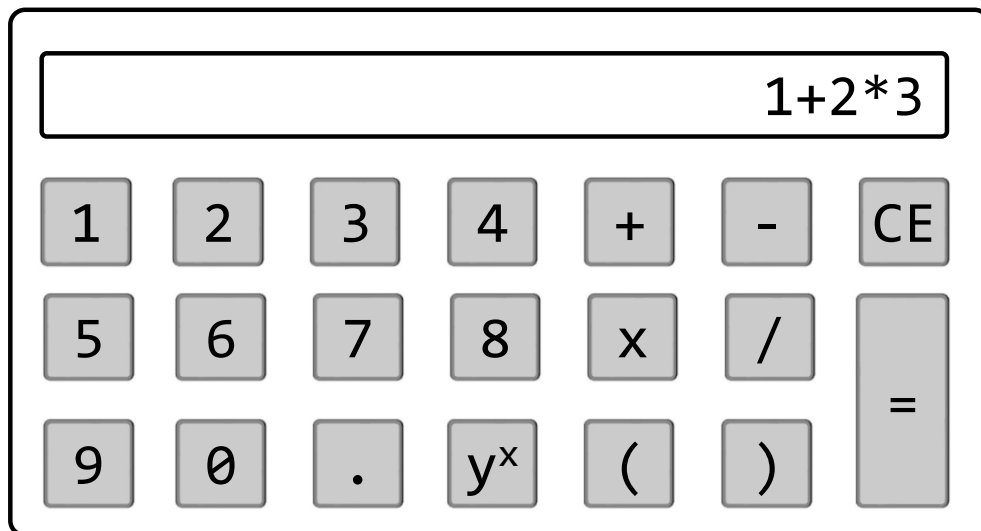
计算器实现分析

徐 辉

xuh@fudan.edu.cn



如何实现一个计算器？



- 功能需求：
 - 操作数支持整数和浮点数，如 0.1、123
 - 运算符支持加、减、乘、除四则运算和指数运算
 - 支持括号

基本思路

- 词法解析：扫描用户输入字符流中的操作数和运算符，将其转化为相应的标签流。
- 句法解析：将标签流按照结合律、分配律、以及括号等四则运算规则进行组织，形成语法解析树。
- 语法制导：将语法解析树翻译成实际的计算代码。

正则表达式声明词法

- 操作数支持整数和浮点数，如 0.1、123
 - 正整数： $[0-9][0-9]^*$
 - 无符号浮点数： $[0-9][0-9]^*(\epsilon|.[0-9][0-9]^*)$
 - 浮点数： $(-|\epsilon)[0-9][0-9]^*(.[0-9][0-9]^*|\epsilon)$
 - 实际情况中，负号（一元运算符）一般不在词法分析环节确定。
 - 需要考虑上下文信息。
- 运算符支持加、减、乘、除四则运算和指数运算
 - +、-、*、/、^
- 支持括号：（、）

优先级和结合性

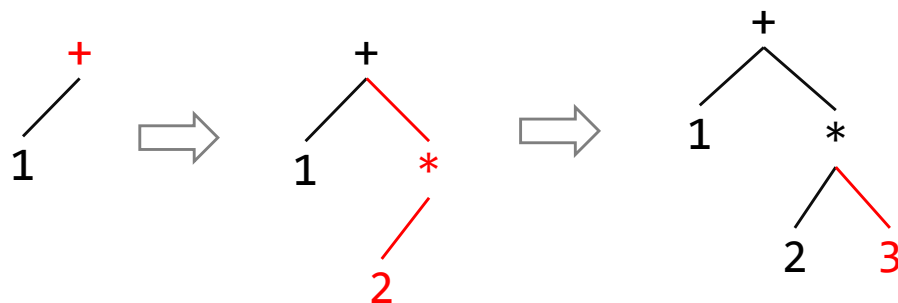
- 优先级 (precedence)
 - 指数运算优先级 > 乘除运算 > 加减运算
- 结合性 (associativity)
 - 加减乘除运算为左结合
 - 指数运算为右结合
 - $2^3^2 = 2^{(3^2)} \neq (2^3)^2$

运算符优先级解析思路

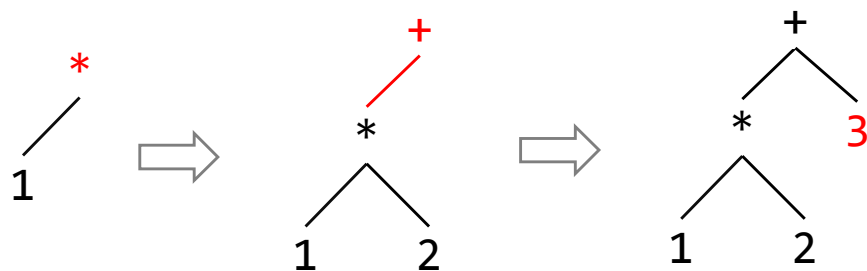
- 优先级爬升算法，按照从左至右的顺序解析并使用栈记录遇到的运算符
 - 如果当前遇到的运算符为左结合
 - 栈顶运算符的优先级大于或等于当前运算符，则基于左结合特性应当将当前运算符作为相应子树的父节点（如何确定），原子树作为其左孩子；
 - 栈顶运算符的优先级低于当前运算符，则当前运算符应作为栈顶运算符的右孩子节点；
 - 如果当前遇到的运算符为右结合
 - 仅有一种运算符（ \wedge ）且优先级最高，应将其作为栈顶运算符的右孩子节点。

示例

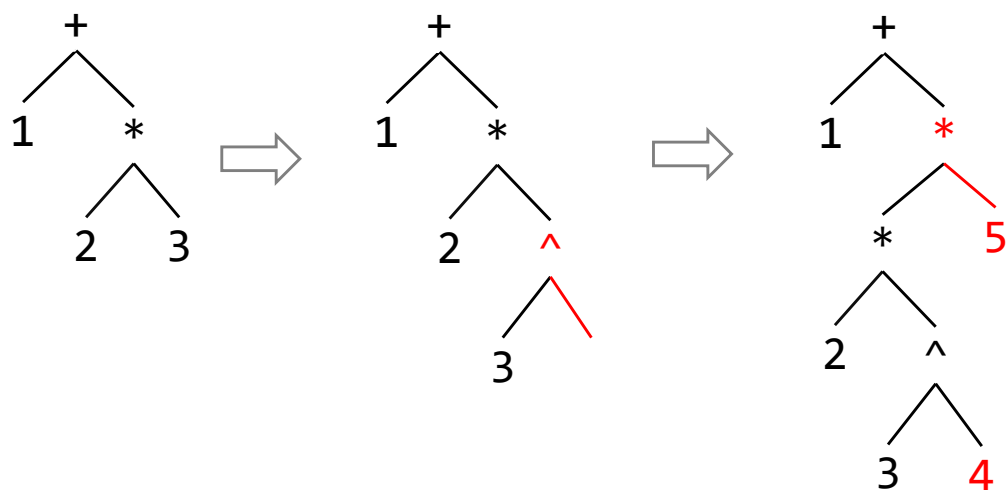
1 + 2 * 3



1 * 2 + 3



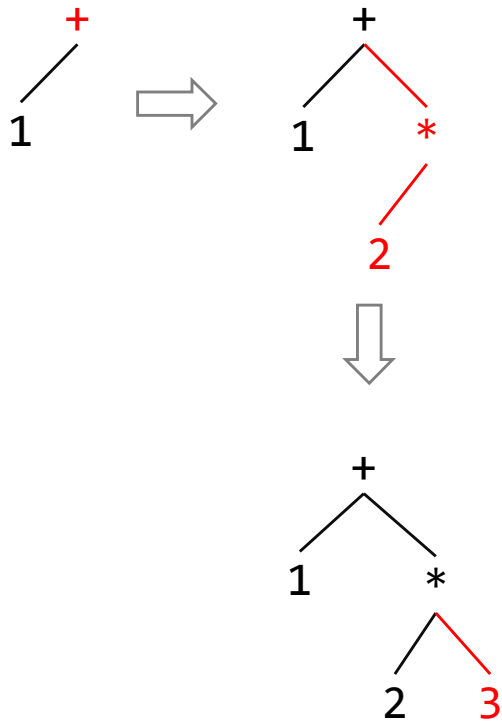
1 + 2 * 3 ^ 4 * 5



示例

初始化优先级 0 1 2 3 4

算式 1 + 2 * 3



```
Pred[ADD] = 1,2
```

```
Pred[SUB] = 1,2
```

```
Pred[MUL] = 3,4
```

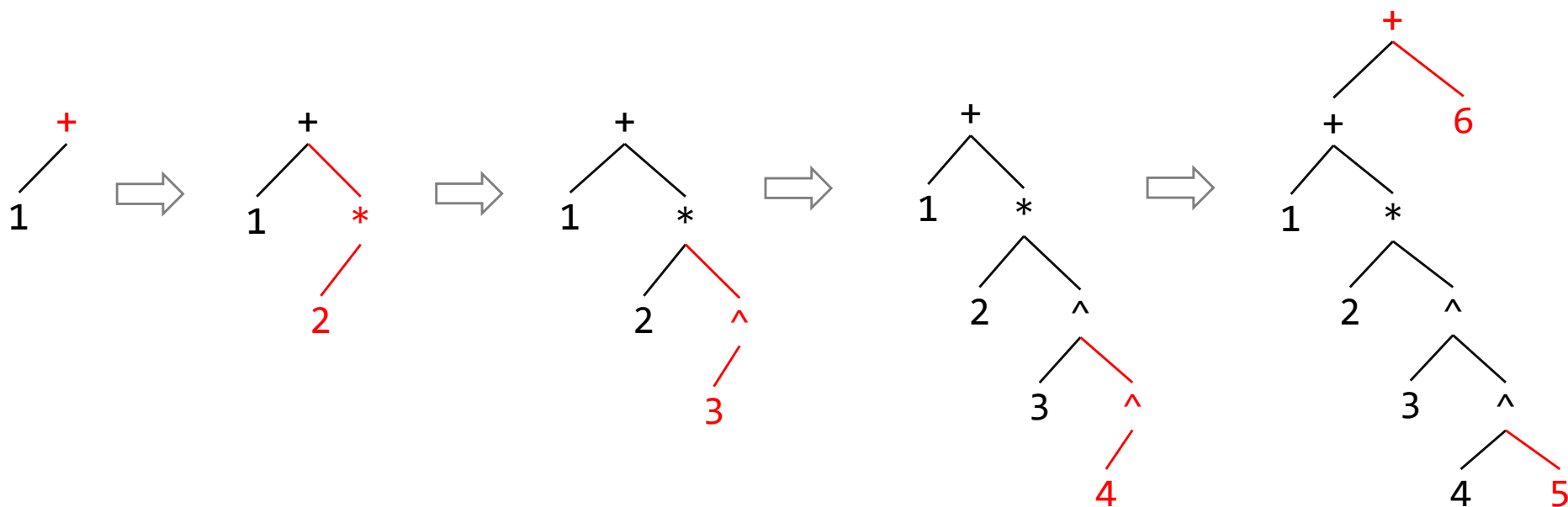
```
Pred[DIV] = 3,4
```

```
Parse(token, precedence) {  
    left = token.next();  
    if left.type != tok::num  
        return -1;  
    while true:  
        op = token.peek();  
        if op.token_type != tok::binop  
            return -1;  
        lp, rp = Pred[op];  
        if lp < precedence  
            break;  
        token.next();  
        right = Parse(token, rp)  
        left = (op, left, right)  
    return left  
}
```


如何支持右结合运算符？

初始化优先级 1 2 3 4 6 5 6 5 1 2

算式 1 + 2 * 3 ^ 4 ^ 5 + 6



制导目标举例：逆波兰表达式

- 波兰表达式：对语法解析树进行先序遍历得到的符号序列
 - $+(+(1*(2^(3^(4\ 5))))))6$
 - 去掉括号无歧义：+ + 1 * 2 ^ 3 ^ 4 5 6
- 逆波兰表达式
 - 对语法解析树进行后序遍历得到的符号序列
 - 1 2 3 4 5 ^ ^ * + 6 +

逆波兰表达式求值

- 按照顺序读取字符串，如果遇到操作数则入栈；
- 如果遇到运算符，则弹出栈顶的两个操作数，求值后将结果如栈；
- 字符串扫描一遍后，栈顶就是表达式的值。