04-F1

栈与队列

中缀表达式求值：问题与构思

邓俊辉

deng@tsinghua.edu.cn

知实而不知名，知名而不知实，皆不知也

应用

❖ 给定语法正确的算术表达式S，计算与之对应的数值

❖ UNIX: $ echo $((0 + (1 + 23) / 4 * 5 * 67 - 8 + 9))

❖ DOS: \> set /a (!0 ^<^< (1 - 2 + 3 * 4)) - 5 * (6 ^| 7) / (8 ^^ 9)

❖ PS: GS> 0 1 23 add 4 div 5 mul 67 mul add 8 sub 9 add =

❖ Excel: = COS(0) + 1 - (2 - POWER((FACT(3) - 4), 5)) * 67 - 8 + 9

❖ Word: = NOT(0) + 12 + 34 * 56 + 7 + 89

❖ calc: 0 ! + 12 + 34 * 56 + 7 + 89 =

❖ calc: 0 ! + 1 - (2 - (3 ! - 4) y 5) * 67 - 8 + 9 =

减而治之

❖ 优先级高的局部执行计算，并被代以其数值

运算符渐少，直至得到最终结果

❖ str(v)：数值v对应的字符串（名）

val(S)：符号串S对应的数值（实）

❖ 设表达式：$S = S_L + S_0 + S_R$

- S_0可优先计算，且
- $val(S_0) = v_0$

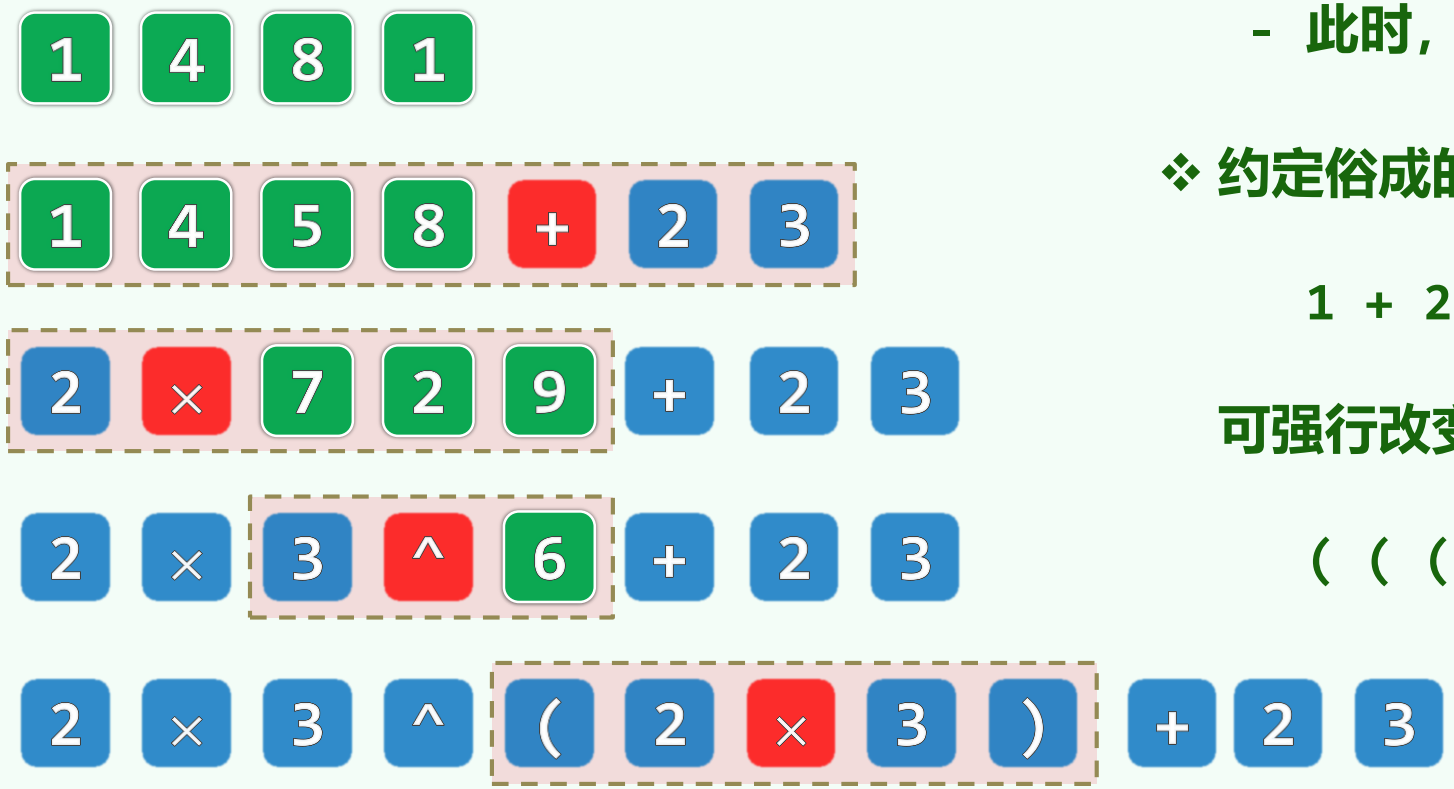
❖ 则有递推化简关系

$$val(S) = val(S_L + str(v_0) + S_R)$$

优先级

❖ 难点：如何高效地找到可优先计算的s_0

（亦即，其对应的运算符）？



❖ 与括号匹配迭代版类似，但亦不尽相同

- 不能简单地按“左先右后”次序处理各运算符
- 此时，需要考虑更多因素...

❖ 约定俗成的优先级：

1 + 2 * 3 ^ 4 !

可强行改变次序的括号：

(((1 + 2) * 3) ^ 4) !

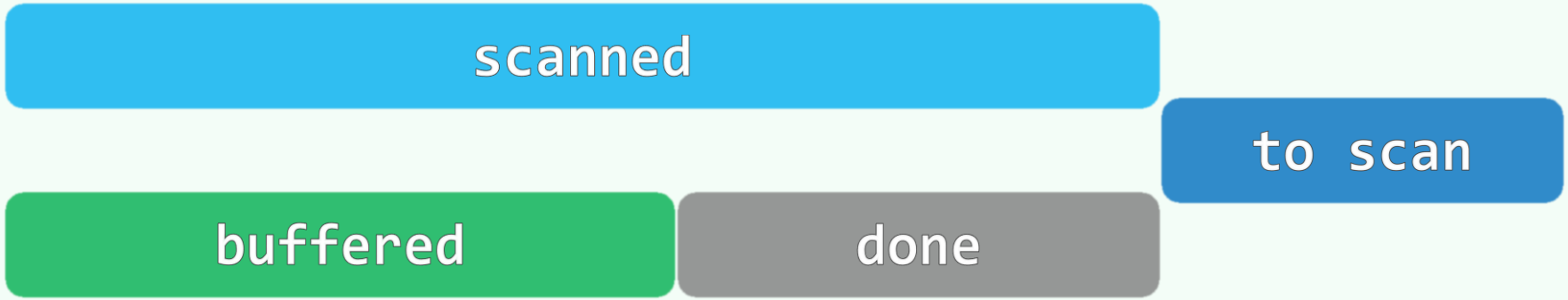
延迟缓冲

❖ 仅根据表达式的前缀，不足以确定各运算符的计算次序

只有获得足够的后续信息，才能确定其中哪些运算符可以执行

❖ 体现在求值算法的流程上

为处理某一前缀，必须提前预读并分析更长的前缀



❖ 为此，需借助某种支持延迟缓冲的机制...

求值算法 = 栈 + 线性扫描

- 自左向右扫描表达式

  用栈记录已扫描的部分，以及中间结果

- 栈内最终所剩的那个元素，即表达式之值

- If（栈的顶部存在可优先计算的子表达式）

  Then 令其退栈并计算；计算结果进栈

  Else 当前字符进栈，转入下一字符