# 1.绪论

# 渐进分析

多项式

Computational problems can be feasibly computed on some computational device only if they can be computed in polynomial time.

- A. Cobham & J. Edmonds

邓俊辉

deng@tsinghua.edu.cn

```
0(1)
```

## ❖常数 (constant function)

//含RAM各基本操作

- 2 = 2015 = 2015 × 2015 = O(1),甚至
- $-2015^{2015}=0(1)$
- ❖ 渐进而言,再大的常数,也要小于递增的变数
- ❖ [General twin prime conjecture, de Polignac 1849]

For every natural number k, there are infinitely many prime pairs p and q such that p - q = 2k

- ❖ [Yitang Zhang, April 2013] k ≤ 35,000,000
- ❖ [Polymath Project, April 2014] k ≤ 123

```
0(1)
```

## ❖ 这类算法的效率最高

## //总不能奢望不劳而获吧

❖ 什么样的代码段对应于常数执行时间?

//应具体分析

#### 一定不含循环?

```
for ( i = 0; i < n; i += n/2015 + 1 );
for ( i = 1; i < n; i = 1 << i ); //log*n,几乎常数
```

#### 一定不含分支转向?

```
if ( (n + m) * (n + m) < 4 * n * m ) goto UNREACHABLE; //不考虑溢出
```

#### 一定不能有(递归)调用?

if 
$$(2 == (n * n) \% 5) 01(n);$$

. . .

# $\mathcal{O}(\log^c n)$

❖ 对数 O(logn)

 $\ln n \mid \lg n \mid \log_{100} n \mid \log_{2015} n$ 

❖ 常底数无所谓

$$\forall$$
 a, b > 0,  $\log_a n = \log_b n = \Theta(\log_b n)$ 

\*常数次幂无所谓

$$\forall$$
 c > 0,  $\log n^c = c \cdot \log n = \Theta(\log n)$ 

❖对数多项式 (poly-log function)

$$123*\log^{321}n + \log^{105}(n^2 - n + 1) = \Theta(\log^{321}n)$$

❖ 这类算法非常有效,复杂度无限接近于常数

$$\forall$$
 c > 0,  $\log n = O(n^c)$ 

```
O(n^c)
```

❖多项式 (polynomial function)

$$100n + 200 = O(n)$$

$$(100n - 500)(20n^2 - 300n + 2015) = O(n \times n^2) = O(n^3)$$

$$(2015n^2 - 20)/(1999n - 1) = O(n^2/n) = O(n)$$
一般地:  $a_k n^k + a_{k-1} n^{k-1} + \ldots + a_1 n + a_0 = O(n^k)$ ,  $a_k > 0$ 

- ❖线性(linear function):所有O(n)类函数
- ❖从Ø(n)到Ø(n²):编程习题主要覆盖的范围
- ❖幂:[  $(n^{2015} 24n^{2009})^{1/3} + 512n^{567} 1978n^{123}$ ]  $^{1/11} = O(n^{61})$
- ❖ 这类算法的效率通常认为已可令人满意,然而...

这个标准是否太低了?

//P难度!