θ1-D1

绪论

复杂度分析：级数

邓俊辉

deng@tsinghua.edu.cn

谁校对时间，谁就会突然老去．

算法分析

❖ **两个主要任务 = 正确性（不变性 x 单调性） + 复杂度**

❖ **为确定后者，真地需要将算法描述为RAM的基本指令，再累计各条代码的执行次数？不必！**

❖ **C++等高级语言的基本指令，均等效于常数条RAM的基本指令**

    在渐近意义下，二者大体相当

    - **分支转向：goto** //算法的灵魂；为结构化而被隐藏了而已

    - **迭代循环：for()、while()、...** //本质上就是 "if + goto"

    - **调用 + 递归（自我调用）** //本质上也是goto

❖ **主要方法： 迭代（级数求和）、递归（递归跟踪 + 递推方程）、实用（猜测 + 验证）**

级数

❖ **算术级数：与末项平方同阶** $T(n) = 1 + 2 + ... + n = \dbinom{n+1}{2} = \dfrac{n(n+1)}{2} = \mathcal{O}(n^2)$

❖ **幂方级数：比幂次高出一阶** $\displaystyle\sum_{k=0}^{n} k^d \approx \int_0^n x^d dx = \left.\dfrac{x^{d+1}}{d+1}\right|_0^n = \dfrac{n^{d+1}}{d+1} = \mathcal{O}(n^{d+1})$

$$T_2(n) = \sum_{k=1}^{n} k^2 = 1^2 + 2^2 + 3^2 + ... + n^2 = n(n+1)(2n+1)/6 = \mathcal{O}(n^3)$$

$$T_3(n) = \sum_{k=1}^{n} k^3 = 1^3 + 2^3 + 3^3 + ... + n^3 = n^2(n+1)^2/4 = \mathcal{O}(n^4)$$

$$T_4(n) = \sum_{k=1}^{n} k^4 = 1^4 + 2^4 + 3^4 + ... + n^4 = n(n+1)(2n+1)(3n^2+3n-1)/30 = \mathcal{O}(n^5)$$

❖ **几何级数：与末项同阶**

$$T_a(n) = \sum_{k=0}^{n} a^k = a^0 + a^1 + a^2 + a^3 + ... + a^n = \dfrac{a^{n+1}-1}{a-1} = \mathcal{O}(a^n), \quad 1 < a$$

$$T_2(n) = \sum_{k=0}^{n} 2^k = 1 + 2 + 4 + 8 + ... + 2^n = 2^{n+1}-1 = \mathcal{O}(2^{n+1}) = \mathcal{O}(2^n)$$

收敛级数

❖ $$\sum_{k=2}^{n} \frac{1}{(k-1) \cdot k} = \frac{1}{1 \cdot 2} + \frac{1}{2 \cdot 3} + \frac{1}{3 \cdot 4} + ... + \frac{1}{(n-1) \cdot n} = 1 - \frac{1}{n} = \mathcal{O}(1)$$

$$\sum_{k=1}^{n} \frac{1}{k^2} = 1 + \frac{1}{2^2} + \frac{1}{3^2} + ... + \frac{1}{n^2} < 1 + \frac{1}{2^2} + \frac{1}{3^2} + ... = \frac{\pi^2}{6} = \mathcal{O}(1)$$

$$\sum_{k \ is \ a \ perfect \ power} \frac{1}{k-1} = \frac{1}{3} + \frac{1}{7} + \frac{1}{8} + \frac{1}{15} + \frac{1}{24} + \frac{1}{26} + \frac{1}{31} + \frac{1}{35} + ... = 1 = \mathcal{O}(1)$$

❖ **几何分布：** $(1 - \lambda) \cdot [1 + 2\lambda + 3\lambda^2 + 4\lambda^3 + ...] = 1/(1-\lambda) = \mathcal{O}(1), \quad 0 < \lambda < 1$

❖ **有必要讨论这类级数吗?**

难道，基本操作次数、存储单元数可能是分数? 是的，某种意义上的确是!

发散级数

❖ **调和级数**： $h(n) = \sum_{k=1}^{n} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{n} = \ln n + \gamma + \mathcal{O}(\frac{1}{2n}) = \Theta(\log n)$

❖ **对数级数**： $\sum_{k=1}^{n} \ln k = \ln \prod_{k=1}^{n} k = \ln n! \approx (n + 0.5) \cdot \ln n - n = \Theta(n \cdot \log n)$

❖ **对数 + 线性 + 指数**： $\sum_{k=1}^{n} k \cdot \log k \approx \int_{1}^{n} x \ln x \, dx = \left. \frac{x^2 \cdot (2 \cdot \ln x - 1)}{4} \right|_{1}^{n} = \mathcal{O}(n^2 \log n)$

$$\sum_{k=1}^{n} k \cdot 2^k = \sum_{k=1}^{n} k \cdot 2^{k+1} - \sum_{k=1}^{n} k \cdot 2^k = \sum_{k=1}^{n+1} (k-1) \cdot 2^k - \sum_{k=1}^{n} k \cdot 2^k$$

$$= n \cdot 2^{n+1} - \sum_{k=1}^{n} 2^k = n \cdot 2^{n+1} - (2^{n+1} - 2) = (n-1) \cdot 2^{n+1} + 2 = \mathcal{O}(n \cdot 2^n)$$

❖ **如有兴趣，不妨读读**： **Concrete Mathematics**   //ex-2.35，Goldbach Theorem