

1-1 求下列函数的渐近表达式：

$3n^2+10n$ ;  $n^2/10+2^n$ ;  $21+1/n$ ;  $\log n^3$ ;  $10\log 3^n$

答：从左到右依次为： $O(n^2)$   $O(2^n)$   $O(1)$   $O(\log n)$   $O(n)$ 。

1-2 试论  $O(1)$ 和  $O(2)$ 的区别。

答：由  $O$  的定义可得  $O(1)=O(2)$ 。用  $O(1)$ 或  $O(2)$ 表示同一个函数时，差别仅在于其中的常数因子。

1-3 按照渐近阶从低到高的顺序排列以下表达式： $4n^2$ ,  $\log n$ ,  $3^n$ ,  $20n$ ,  $2$ ,  $n^{2/3}$ 。又  $n!$ 应该排在哪一位？

答：顺序为： $2$ ,  $\log n$ ,  $n^{2/3}$ ,  $20n$ ,  $4n^2$ ,  $3^n$ ,  $n!$ 。 $n!$ 排在最后。

1-4 (1) 假设某算法在输入规模为  $n$  时的计算时间为  $T(n)=3 \times 2^n$ 。在某台计算机上实现并完成该算法的时间为  $t$  秒。现有另一台计算机，其运行速度为第一台的 64 倍，那么在这台新机器上用同一算法在  $t$  秒内能解输入规模为多大的问题？

(2) 若上述算法的计算时间改进为  $T(n)=n^2$ ，其余条件不变，则在新机器上用  $t$  秒时间能解输入规模为多大的问题？

(3) 若上述算法的计算时间进一步改进为  $T(n)=8$ ，其余条件不变，那么在新机器上用  $t$  秒时间能解输入规模为多大的问题？

答：

(1) 设新机器用同一算法在秒内能解输入规模为  $n_1$  的问题。因此有： $t=3 \times 2^{n_1} - (3 \times 2^n)/64$ 。

解得  $n_1=n+6$ 。

(2)  $n_1^2 - 64n^2 \rightarrow n_1=8n$ 。

(3) 由于  $T(n)=$ 常数，因此算法可解任意规模的问题。

1-5 硬件厂商 XYZ 公司宣称他们最新研制的微处理器运行速度为其竞争对手 ABC 公司同类产品的 100 倍。对于计算复杂性分别为  $n$ 、 $n^2$ 、 $n^3$  和  $n!$  的各算法，若用 ABC 公司的计算机在 1 小时内能解输入规模为  $n$  的问题，那么用 XYZ 公司的计算机在 1 小时内分别能解输入规模为多大的问题？

答：设 XYZ 公司输入规模为  $n'$

$$n'=100n$$

$$n'^2=100n^2 \rightarrow n'=10n$$

$$n'^3=100n^3 \rightarrow n'=(100n)^{1/3}=4.64n$$

$$n!=100n! \rightarrow n'<n+\log 100=n+6.64$$

1-6 对于下列各组函数  $f(n)$  和  $g(n)$ , 确定  $f(n)=O(g(n))$  或  $f(n)=\Omega(g(n))$  或  $f(n)=\theta(g(n))$ , 并简述理由。

- |                        |                 |                       |                |
|------------------------|-----------------|-----------------------|----------------|
| (1) $f(n)=\log n^2$ ;  | $g(n)=\log n+5$ | (5) $f(n)=10$ ;       | $g(n)=\log 10$ |
| (2) $f(n)=\log n^2$ ;  | $g(n)=\sqrt{n}$ | (6) $f(n)=\log^2 n$ ; | $g(n)=\log n$  |
| (3) $f(n)=n$ ;         | $g(n)=\log^2 n$ | (7) $f(n)=2^n$ ;      | $g(n)=100n^2$  |
| (4) $f(n)=n\log n+n$ ; | $g(n)=\log n$   | (8) $f(n)=2^n$ ;      | $g(n)=3^n$     |

答: (1)  $\log n^2 = \theta(\log n + 5)$     (2)  $\log n^2 = O(\sqrt{n})$   
 (3)  $n = \Omega(\log^2 n)$     (4)  $n\log n + n = \Omega(\log n)$   
 (5)  $10 = \theta(\log 10)$     (6)  $\log^2 n = \Omega(\log n)$   
 (7)  $2^n = \Omega(100n^2)$     (8)  $2^n = O(3^n)$

1-7 证明  $n! = o(n^n)$ 。

$$\therefore n! = \sqrt{2\pi n} \left(\frac{n}{e}\right)^n \left[1 + \theta\left(\frac{1}{n}\right)\right]$$

$$\therefore \lim_{n \rightarrow \infty} \frac{n!}{n^n} = \frac{\sqrt{2\pi n} \left[1 + \theta\left(\frac{1}{n}\right)\right]}{e^n} = 0$$

1-8 下面的算法段用于确定  $n$  的初始值。试分析该算法段所需计算时间的上界和下界。

```
while(n>1)
    if(odd(n))
        n = 3*n+1;
    else
        n = n/2;
```

答: 该算法表述的是著名的  $3n+1$  问题。在最坏情况下, 该算法的计算时间下界显然为  $\Omega(\log n)$ 。

算法的计算时间上界至今未知。

1-9 证明: 如果一个算法在平均情况下的计算时间复杂性为  $\theta(f(n))$ , 则该算法在最坏情况下所需的计算时间为  $\Omega(f(n))$ 。

$$\therefore T_{avg}(N) = \sum_{I \in D_N} P(I) T(N, I) \leq \sum_{I \in D_N} P(I) \max_{I' \in D_N} T(N, I')$$

$$= T(N, I^*) \sum_{I \in D_N} P(I) = T(N, I^*) = T_{max}(N)$$

$$\therefore T_{max}(N) = \Omega(T_{avg}(N)) = \Omega(\theta(f(n))) = \Omega(f(n))$$