# 《计算机算法设计与分析》第一章作业

1-1 求下列函数的渐进表达式：

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

答：$O(n^2)$ ; $O(2^n)$ ; $O(1)$ ; $O(\log n)$ ; $O(n)$

1-2 试论 O(1)和 O(2)的区别。

答：由 O 的定义可知，O(1)=O(2)，用 O(1)或 O(2)表示同一个函数时，差别仅在于常数因子 c。

1-3 按照渐进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：从低到高排列为：$2 < \log n < n^{2/3} < 20n < 4n^2 < 3^n < n!$ 。

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。现拥有另一台计算机，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，则在新机器上用 t 秒时间能解输入规模为多大的问题？

答：

(1) $3\times 2^n \times 64 = 3\times 2^x$ 解得 $X=n+6$，能解 n+6 规模的问题。

(2) $64n^2 = X^2$ 解得 X=8n，能解 8n 规模的问题。

(3) T(n)为常数，算法可以解任意规模的问题。

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 小时内分别能解规模为多大的问题？

答：

对于复杂性为 n 的算法，X=100n，能解规模 100n 的问题；

对于复杂性为 $n^2$ 的算法，$X^2$=100 $n^2$，能解规模 10n 的问题；

对于复杂性为 $n^3$ 的算法，$X^3$=100 $n^3$，能解规模 4.64n 的问题；

对于复杂性为 n！的算法，X！=100n！，能解规模 n+log100 即 n+6.64 的问题；

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)= Ω(g(n))或 f(n)= θ(g(n))，并简述理由。

(1) f(n)=$\log n^2$ g(n)=logn+5

(2) f(n)= $\log n^2$ g(n)= $\sqrt{n}$

(3) f(n)=n g(n)= $\log^2 n$

(4) f(n)=nlogn+n g(n)=logn

(5) f(n)=10 g(n)=log10

(6) f(n)= $\log^2 n$ g(n)=logn

(7) f(n)=$2^n$ g(n)=100$n^2$

(8) f(n)= $2^n$ g(n)=$3^n$

答：

(1) $f(n)=\log n^2=\theta(\log n+5)$ ，同阶

(2) $f(n)=\log n^2=O(\sqrt{n})$ ，f(n)的阶不高于 g(n)的阶

(3) $f(n)=n=\Omega(\log^2 n)$ ，f(n)的阶不低于 g(n)的阶

(4) $f(n)=n\log n+n=\Omega(\log n)$ ，f(n)的阶不低于 g(n)的阶

(5) $f(n)=10=\theta(\log 10)$ ，同阶

(6) $f(n)=\log^2 n=\Omega(\log n)$ ，f(n)的阶不低于 g(n)的阶

(7) $f(n)=2^n=\Omega(100n^2)$ ，f(n)的阶不低于 g(n)的阶

(8) $f(n)=2^n=O(3^n)$ ，f(n)的阶不高于 g(n)的阶

1-7 证明 n！=$O(n^n)$。

证明：

当 n 趋近于无穷时，$\lim n!/n^n=(1/n)(2/n)(3/n)\cdots(n/n)=0$，即 n!的阶不高于 $n^n$ 的阶，n！=$O(n^n)$。

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：在最坏情况下，该算法的计算时间下界为Ω(logn)，无法分析计算时间的上界。

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

答：由最坏情况和平均情况下的时间复杂性在数学上的定义可知：

$T_{avg}(N)= \sum P(I)T(N,I) <= \sum P(I)\max T(N,I')$

$=T(N,I^*)\sum P(I)$

$=T(N,I^*)$

$=T_{max}(N)$

其中，$I^*$ 是 $D_N$ 中使 $T(N,I^*)$达到 $T_{max}(N)$的合法输入，而 P(I)是在算法的应用中出现输入 I 的概率。

因此 $T_{max}(N)=\Omega(T_{avg}(N))=\Omega(\theta(f(n)))=\Omega(f(n))$，证毕。