Θ theta 精（紧）确界 =

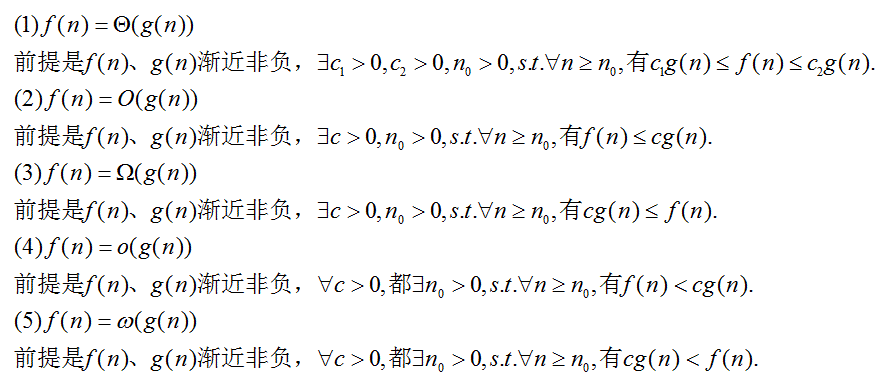
O 大O 上界 <=

o 非紧 上界 <

Ώ (大omega) 下界 >=

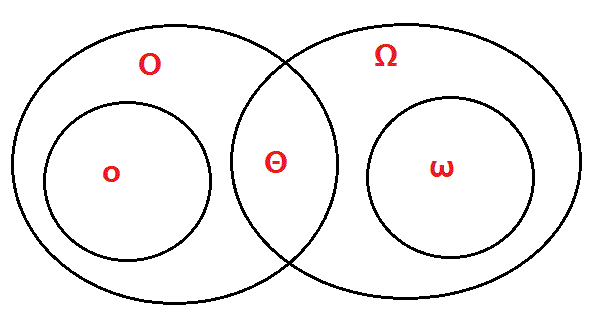
ω omega 非紧的下界 >

符号定义：



这些定义的前提是 f(n) g(n) 渐进非负的，当n趋近于无穷大时，f(n) g(n)都非负。

4，5条 定义，任意的c。



从上面的图可以看出：

(1)如果f(n)=Θ(g(n)),则f(n)=O(g(n))且f(n)=Ω(g(n))。

(2)如果f(n)= o (g(n)),则f(n)=O(g(n))。

(3)如果f(n)=ω(g(n))，则f(n)=Ω(g(n))。

(4)如果f(n)=O(g(n)),则要么是f(n)= o (g(n))，要么是f(n)=Θ(g(n))。

(5)如果 f(n)=Ω(g(n)) ,则要么是f(n)=ω(g(n))，要么是f(n)=Θ(g(n))。

原因其实很简单因为：

(1)如果f(n)=Θ(g(n))，则根据定义一定存在c1,c2,n0，使得对于任意的n>=n0，都有c1g(n)<=f(n)<=c2g(n)，因此必定存在

        "c=c2,n0,使得对于任意的n>=n0,都有f(n)<=cg(n)"

        "c=c1,n0,使得对于任意的n>=n0,都有cg(n)<=f(n)"

(2)如果f(n)= o (g(n)),则根据定义一定对于任意的c，都存在n0，使得任意的n>=n0，都有f(n)<cg(n)，因此必定存在“存在c,n0，使得对于任意的n>=n0,都有f(n)<=cg(n)”

(3)如果f(n)=  ω (g(n)),则根据定义一定对于任意的c，都存在n0，使得任意的n>=n0，都有cg(n)<f(n)，因此必定存在“存在c,n0，使得对于任意的n>=n0,都有cg(n)<=f(n)”

了解了这些定义后，给出一个概念：这些渐近记号表示的都是集合，比如 O (n^2)表示的是一个集合，可以是n,1,n^2等，

因此对于这些渐近记号的使用最准确应该是“f(n)∈ O (g(n))”，但是一般都是写成“f(n)=O(g(n))”。

给出一些例子：

O(n^2)可以是n,2n,1,2n^2等。

Θ(n^2)可以是n^2,3n^2等。

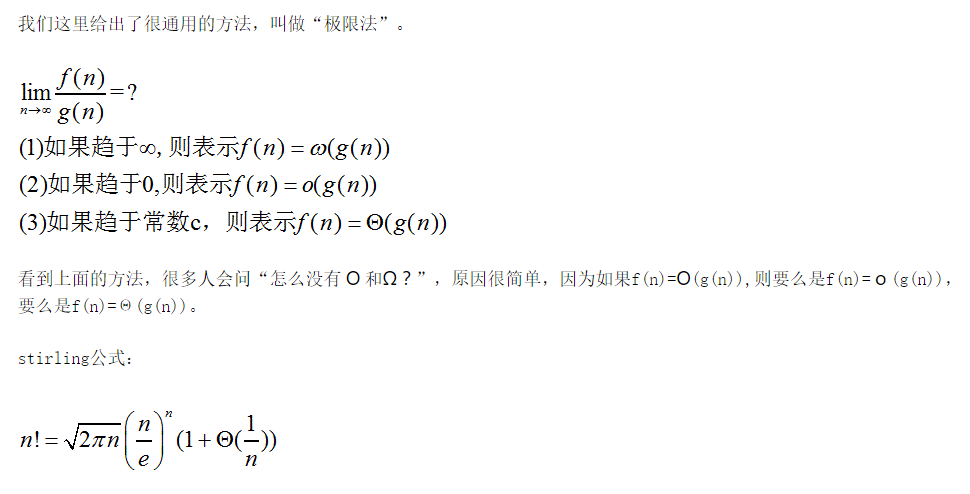
ω(n^2)可以是n^3,n^10等,但不能是n^2。

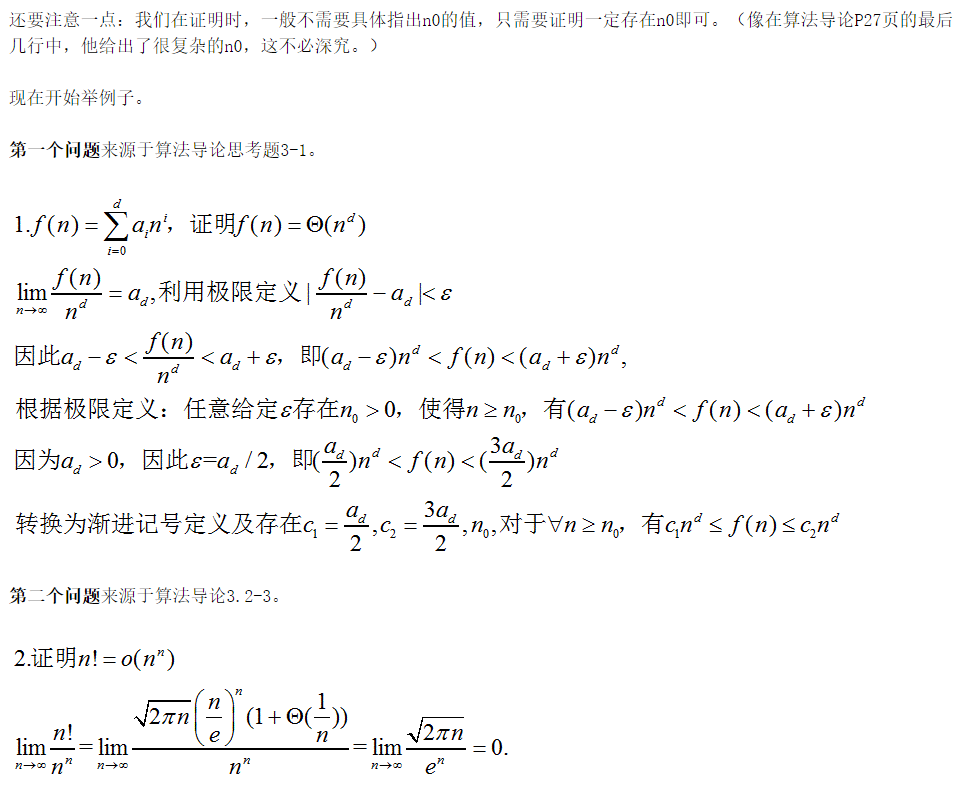
Ω(n^2)可以是n^2,n^3,n^10等。

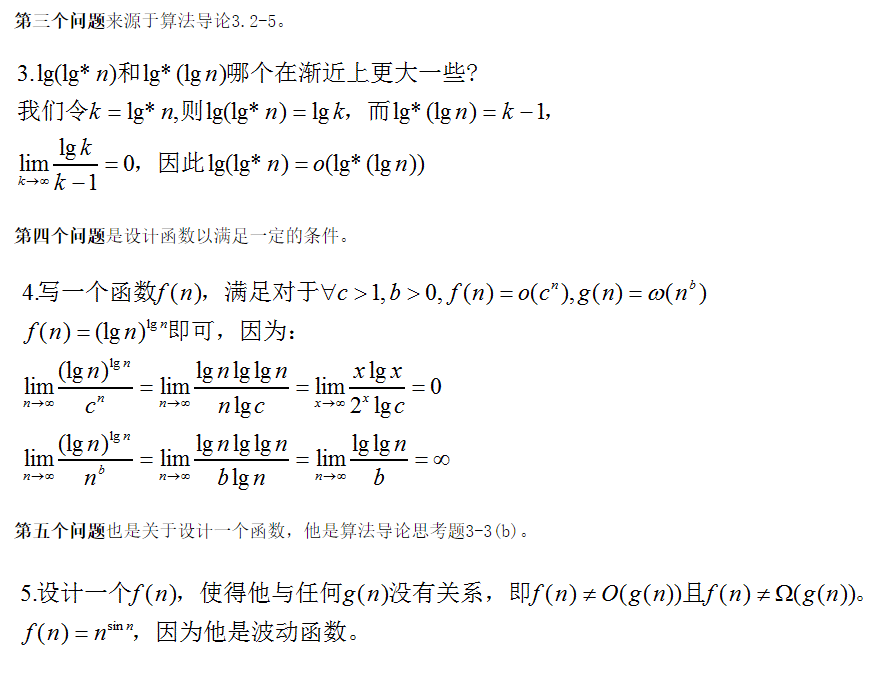
o(n^2)可以是n,1,3n等,但不能是n^2。

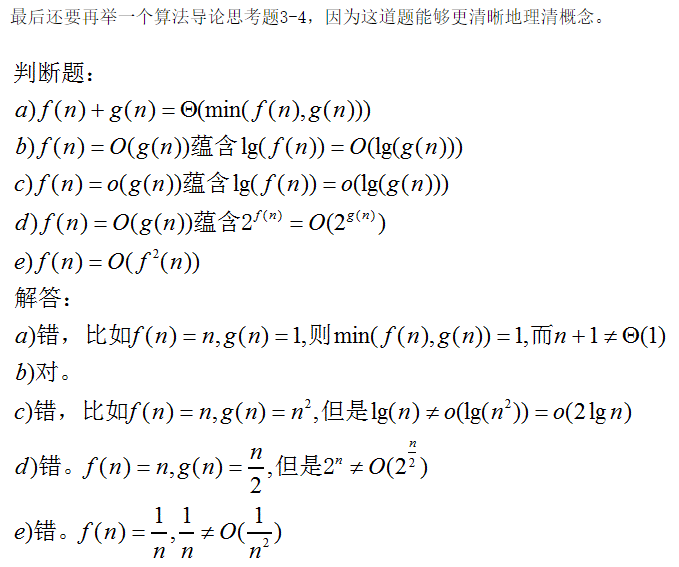
一般我们对于算法复杂度的描述都是用 O 记号，比如BellmanFord的复杂度为O(VE)，表示对于所有的输入，都满足O(VE)。

**二、判断两个函数的渐近关系**









空间复杂度

时间复杂度