运用渐进分析计算程序的时间复杂度

例1. 求n个数a1, ..., an中最大数的问题。

```
max ← a1
for i = 2 to n {
    if (ai > max)
        max ← ai
}
```

例1. 求n个数a1, …, an中最大数的问题。

```
max ← a1
for i = 2 to n {
    if (ai > max)
        max ← ai
}
```

时间复杂度:O(n)

例2. 求平面上n个点(x1, y1), …, (xn, yn)中最近两个点之间的距离.

对每对点都尝试一下。

```
min ← (x1 - x2)^2 + (y1 - y2)^2
for i = 1 to n {
    for j = i+1 to n {
        d ← (xi - xj)^2 + (yi - yj)^2
        if (d < min)
            min ← d
    }
}
```

例2. 求平面上n个点(x1, y1), ..., (xn, yn)中最近两个点之间的距离.

对每对点都尝试一下可得出平方时间的算法。

```
min ← (x1 - x2)^2 + (y1 - y2)^2
for i = 1 to n {
    for j = i+1 to n {
        d ← (xi - xj)^2 + (yi - yj)^2
        if (d < min)
            min ← d
    }
}
```

时间复杂度 $O(n^2)$

电子科技大学

例3. 计算平面上两点间的最短距离。看下面这段程序的时间复杂度

```
d[i*j]
for i = 1 to n {
    for j = i+1 to n {
        d[k] ← (xi - xj)² + (yi - yj)²
    }
}
a=d[1]
for i = 1 to i*j {
    if (d[i] < a)
        a ← d[i]
}
```

例3. 计算平面上两点间的最短距离。看下面这段程序的时间复杂度

```
d[i*j]
for i = 1 to n {
    for j = i+1 to n {
        d[k] ← (xi - xj)² + (yi - yj)²
    }
}
a=d[1]
for i = 1 to i*j {
    if (d[i] < a)
        a ← d[i]
}
```

时间复杂度 $O(n^2+n)=O(n^2)$

例4. K元素独立集问题 给出一个图，是否存在一个具有k个顶点的子图，该子图中任意两点间无边存在。

穷举搜索产生运行时间的算法 .

```
foreach subset S of k nodes {
   check whether S in an independent
set
   if (S is an independent set)
      report S is an independent set
   }
}
```

例4. K元素独立集问题 给出一个图，是否存在一个具有k个顶点的子图，该子图中任意两点间无边存在。
穷举搜索产生运行时间的算法 .

```
foreach subset S of k nodes {
   check whether S in an independent
set
   if (S is an independent set)
      report S is an independent set
   }
}
```

检测点集S中不存在一条边 = $O(k^2)$.
选出1-k个元素子集 $= \binom{n}{k} = \frac{n\,(n-1)\,(n-2)\cdots(n-k+1)}{k\,(k-1)\,(k-2)\cdots(2)\,(1)} \leq \frac{n^k}{k!}$

$O(k^2\,n^k / k!) = O(n^k)$.