提纲

算法的由来

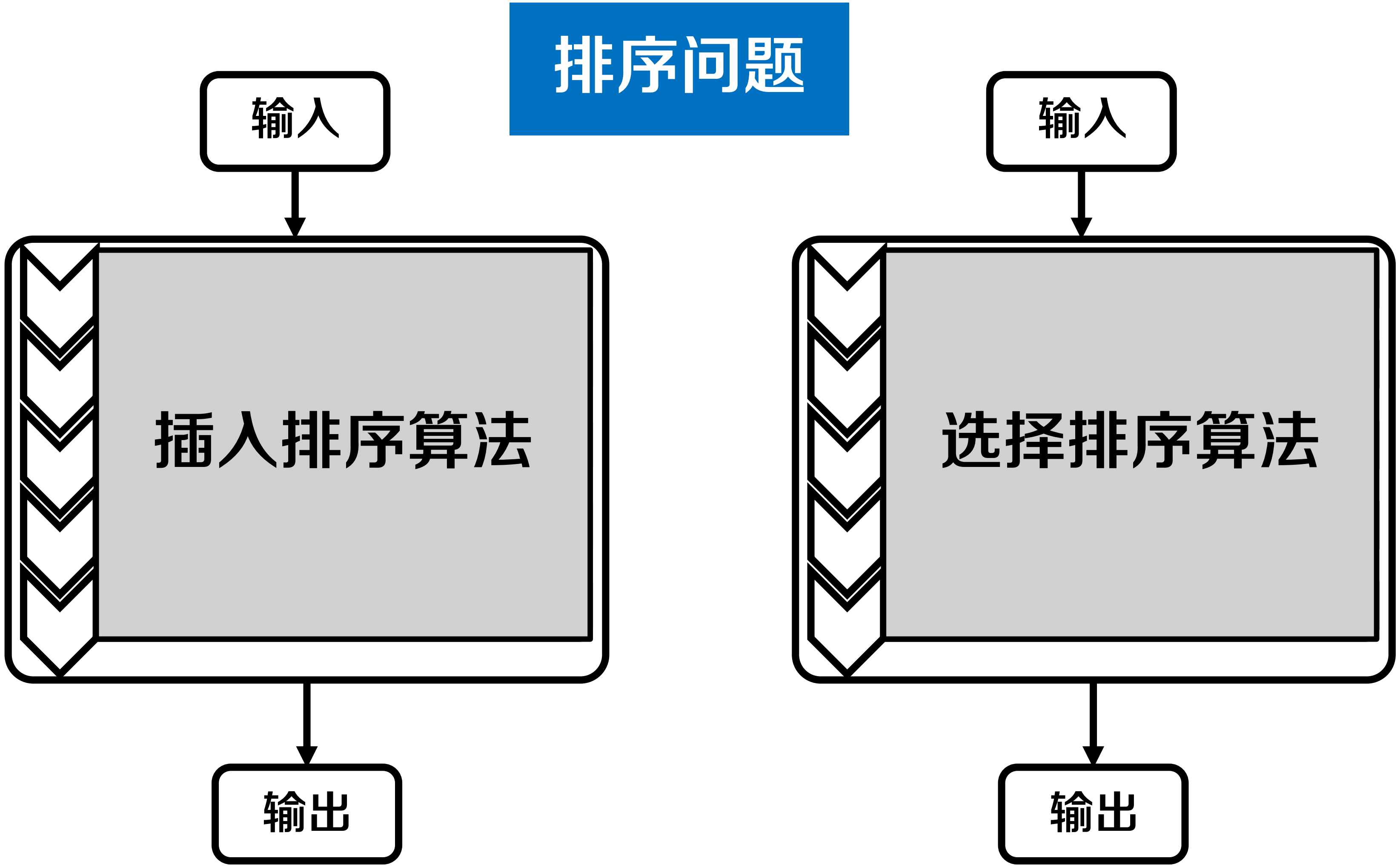
算法的定义

算法的性质

算法的表示

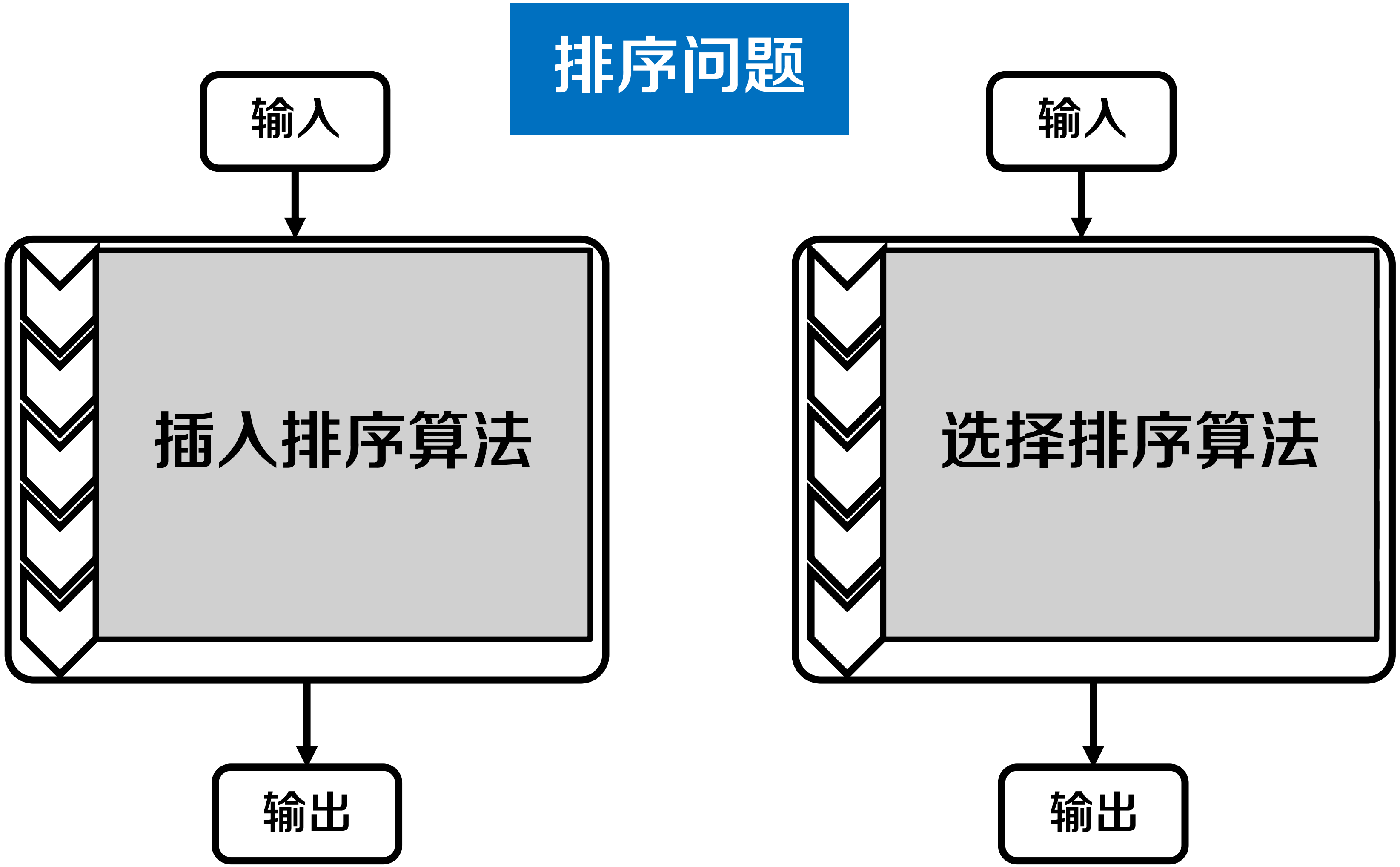
算法的分析

回顾



问题：如何比较不同算法性能?

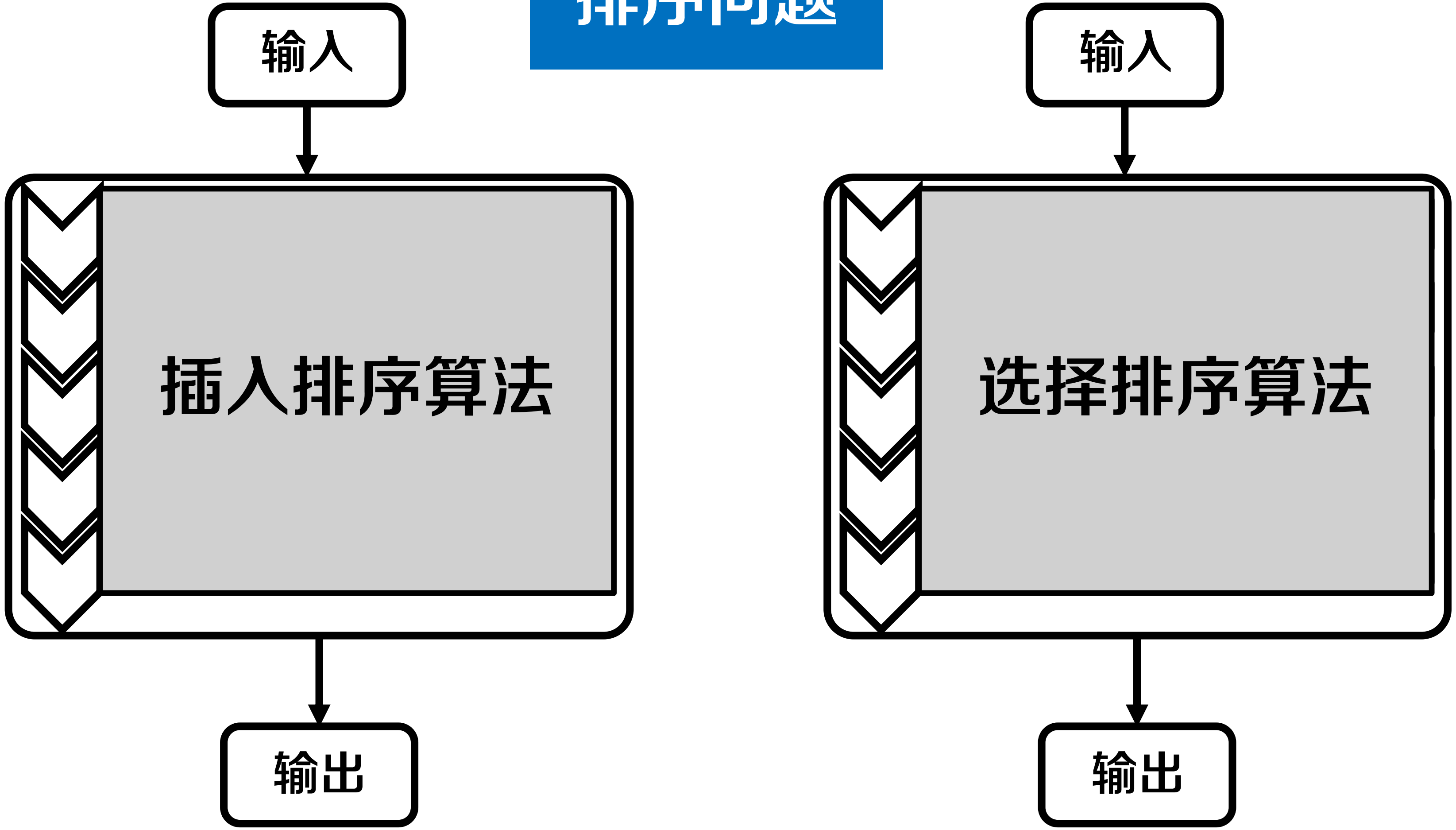
回顾



分析算法的运行时间

回顾

排序问题



算法分析的原则

算法分析的工具

算法分析的实例

分析算法的运行时间

算法分析的原则

- 机器的运算速度影响算法的运行时间

机器	运算速度	运行算法	运行时间
天河三号	百亿亿次/秒	插入排序	无法公平比较
个人电脑	十亿次/秒	选择排序	

算法分析的原则

- 机器的运算速度影响算法的运行时间

<table>
<tr><th>机器</th><th>运算速度</th><th>运行算法</th><th>运行时间</th></tr>
<tr><td>天河三号</td><td>百亿亿次/秒</td><td>插入排序</td><td rowspan="2">无法公平比较</td></tr>
<tr><td>个人电脑</td><td>十亿次/秒</td><td>选择排序</td></tr>
</table>

分析算法的运行时间应独立于机器

算法分析的原则

- 归纳基本操作
  - 如：运算、赋值、比较

$+$	$-$	$\times$	$\div$
$:=$	$>$	$<$	$=$

算法分析的原则

- 归纳基本操作
  - 如：运算、赋值、比较

$+$	$-$	$\times$	$\div$
$:=$	$>$	$<$	$=$

- 统一机器性能
  - 假设基本操作代价均为1

算法分析的原则

- 归纳基本操作
  - 如：运算、赋值、比较

+	−	×	÷
:=	>	<	=

- 统一机器性能
  - 假设基本操作代价均为1


统一机器性能后，算法运行时间依赖于问题输入规模与实例

算法分析的原则

- **相同输入规模，实例影响运行**

输入: 数组$A[a_1, a_2, ..., a_n]$
输出: 升序数组$A'[a'_1, a'_2, ..., a'_n]$，满足 $a'_1 \le a'_2 \le ... \le a'_n$

```
for j ← 2 to n do
    key ← A[j]
    i ← j − 1
    while i > 0 and A[i] > key do
        A[i + 1] ← A[i]
        i ← i − 1
    end
    A[i + 1] ← key
end
```

循环次数未知

插入排序算法伪代码

算法分析的原则

- 相同输入规模，实例影响运行
  - 插入排序最好情况：数组升序
    - 比较次数：$\underbrace{1+1+1+\cdots+1}_{n-1} = n-1$

4	8	13	14	17	18	21	22	24	28	32	37	40	40	47	48

算法分析的原则

- 相同输入规模，实例影响运行
  - 插入排序最好情况：数组升序
    - 比较次数：$\underbrace{1+1+1+\cdots+1}_{n-1} = n-1$

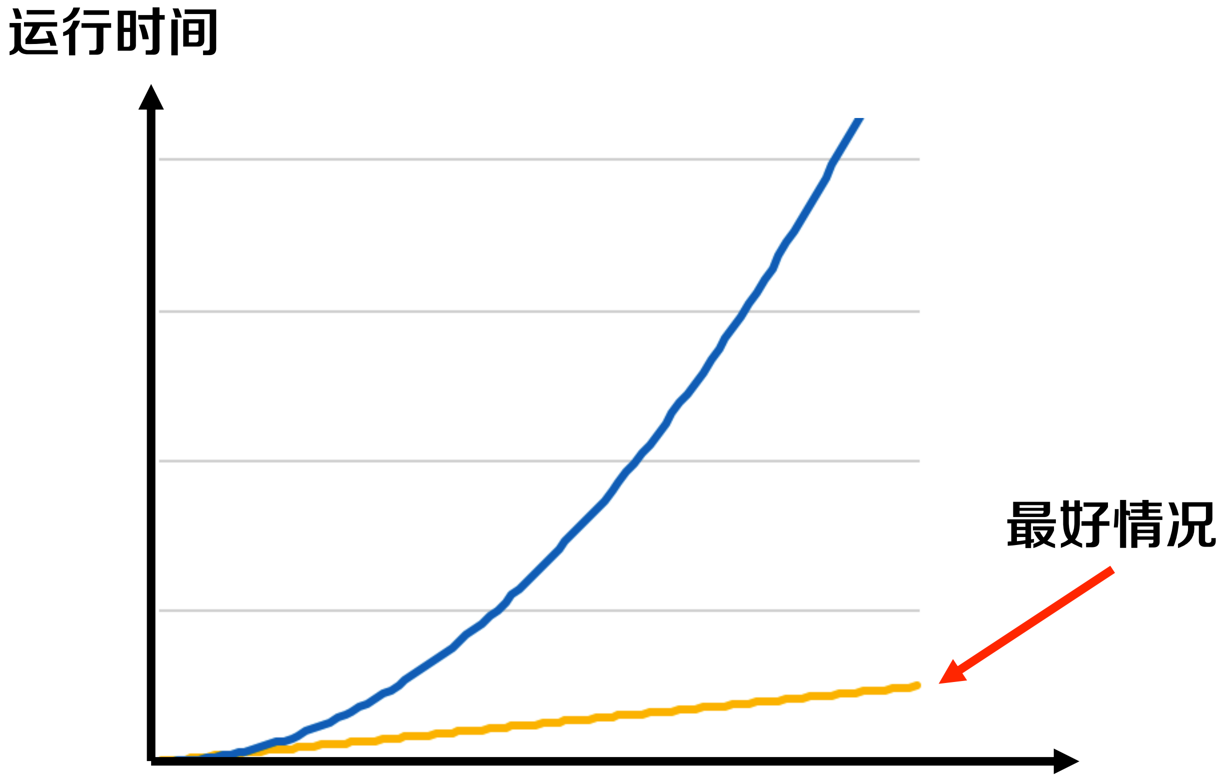
4	8	13	14	17	18	21	22	24	28	32	37	40	40	47	48

  - 插入排序最坏情况：数组降序
    - 比较次数：$1+2+3+\cdots+(n-1) = \frac{n(n-1)}{2}$

48	47	40	40	37	32	28	24	22	21	18	17	14	13	8	4

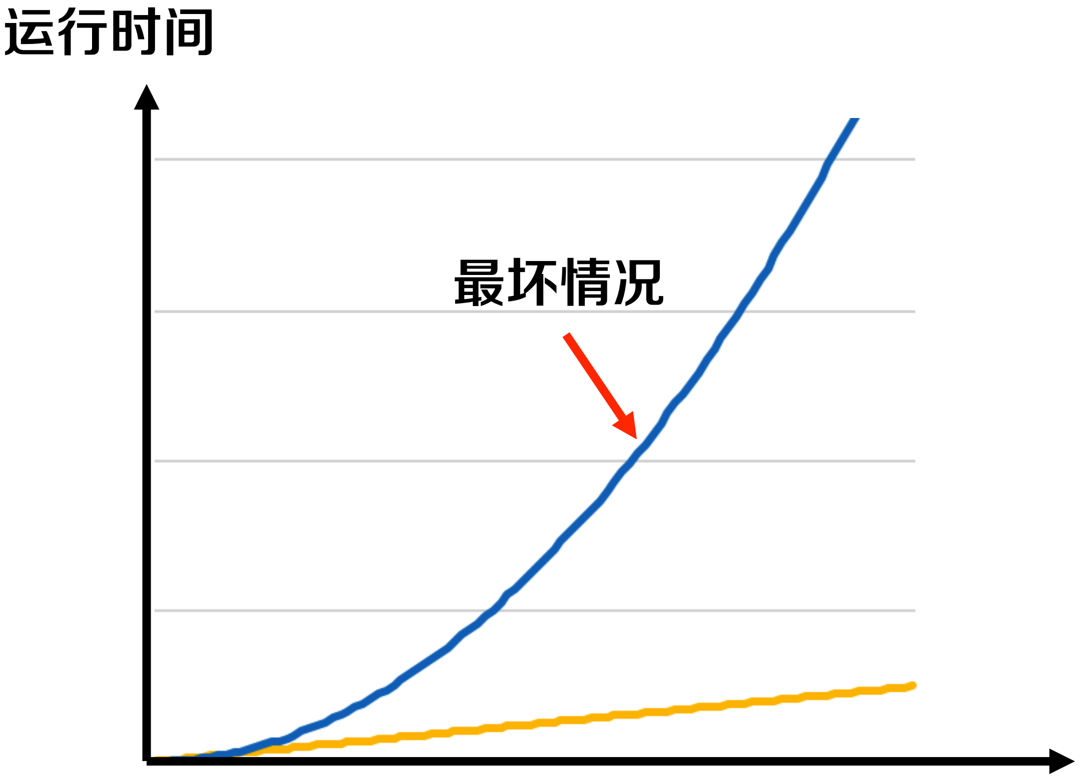
算法分析的原则

输入情况	情况说明
最好情况	不常出现，不具普遍性



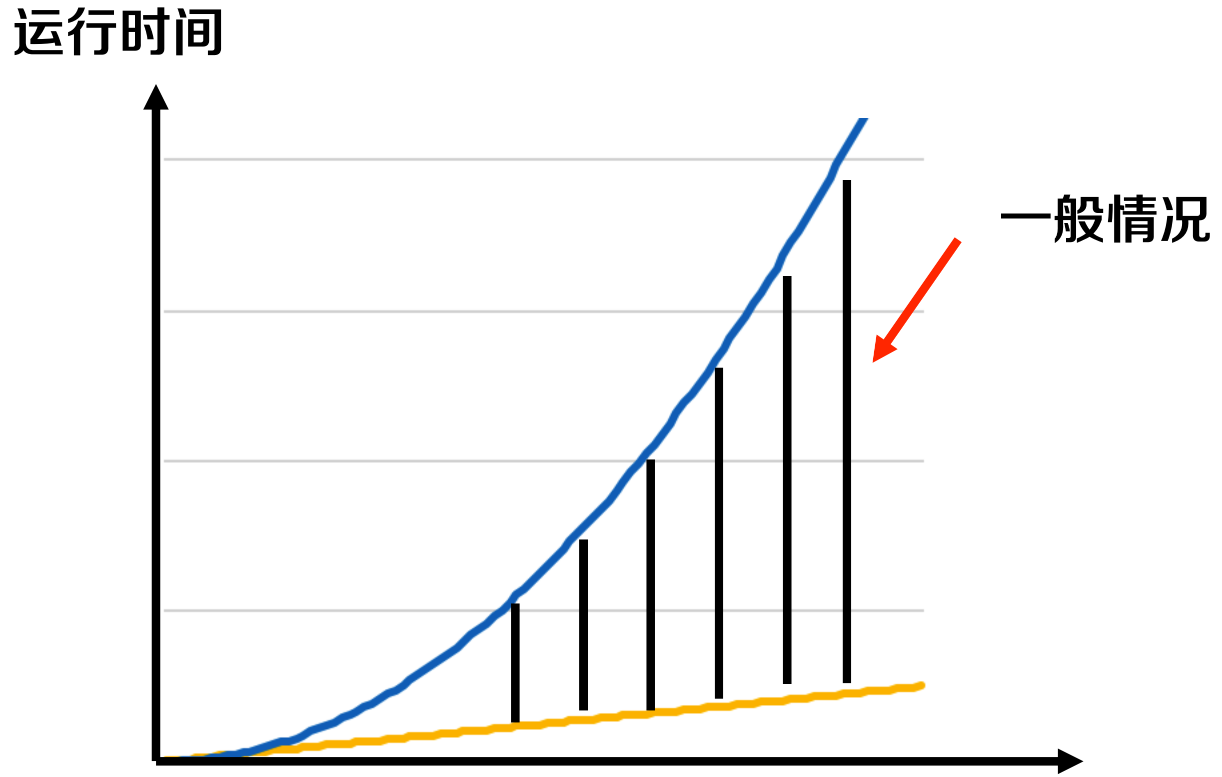
算法分析的原则

输入情况	情况说明
最好情况	不常出现，不具普遍性
最坏情况	确定上界，更具一般性



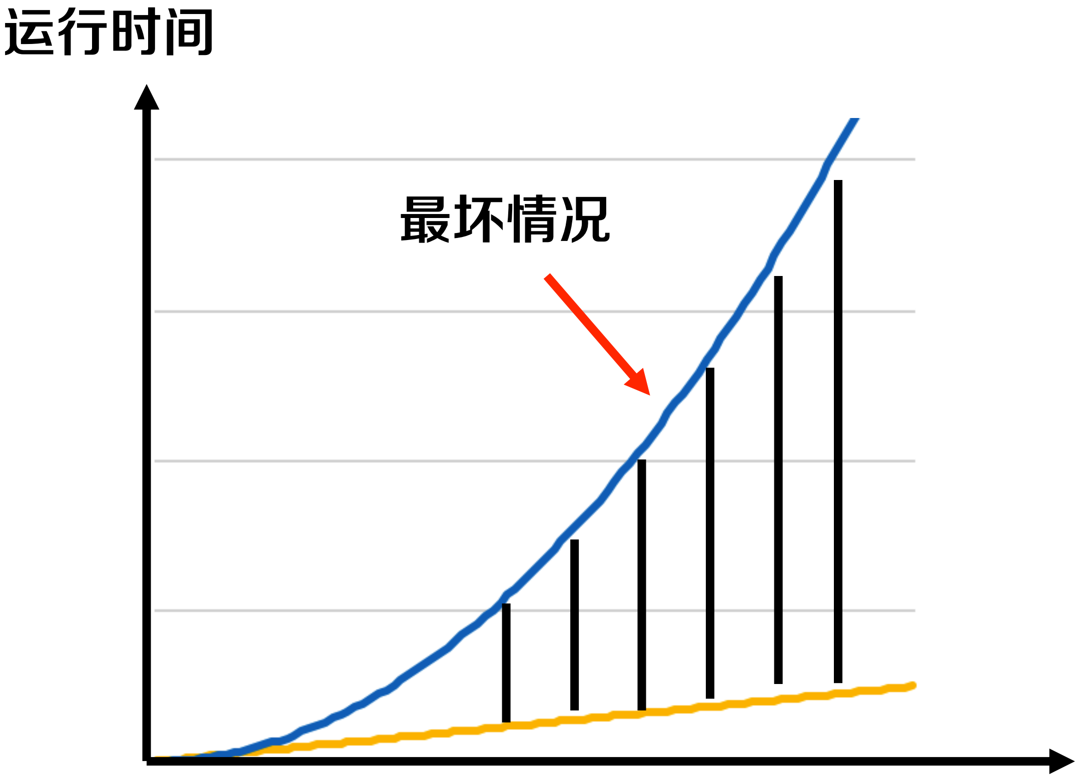
算法分析的原则

输入情况	情况说明
最好情况	不常出现，不具普遍性
最坏情况	确定上界，更具一般性
一般情况	情况复杂，分析难度大



算法分析的原则

输入情况	情况说明
最好情况	不常出现，不具普遍性
最坏情况	确定上界，更具一般性
一般情况	情况复杂，分析难度大



常用最坏情况分析算法运行时间

算法分析的原则

仅依赖于输入规模，求 T(n) 来分析算法时间效率

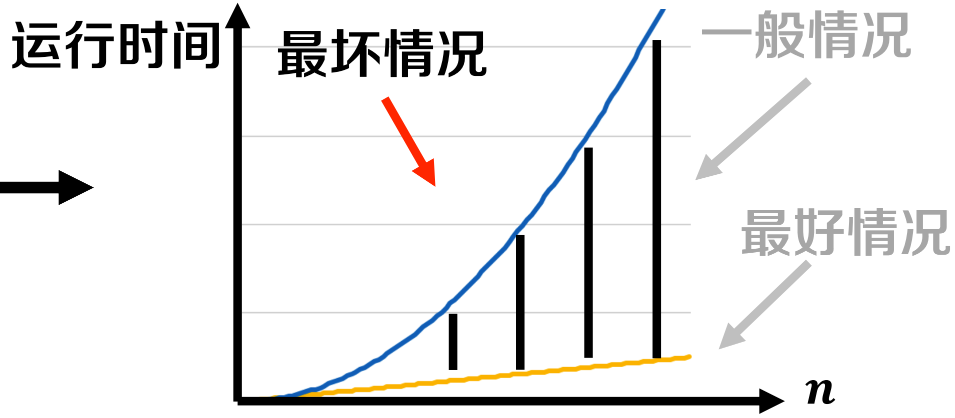
- ①统一机器性能

+	−	×	÷
:=	>	<	=

- ②分析最坏情况

4	···	40	47	48

48	···	13	8	4



算法运行时间仅依赖于问题输入规模n，表示为$T(n)$

算法分析的工具

- **插入排序最坏情况**

```
for j ← 2 to n do
    key ← A[j]
    i ← j − 1
    while i > 0 and A[i] > key do
        A[i + 1] ← A[i]
        i ← i − 1
    end
    A[i + 1] ← key
end
```

- **选择排序最坏情况**

```
for i ← 1 to n − 1 do
    for j ← i + 1 to n do
        if A[i] > A[j] then
            交换 A[i] 和 A[j]
        end
    end
end
```

算法分析的工具

- 插入排序最坏情况

```
for j ← 2 to n do
    key ← A[j]
    i ← j − 1
    while i > 0 and A[i] > key do
        A[i + 1]← A[i]
        i ← i − 1
    end
    A[i + 1] ← key
end
```

- 选择排序最坏情况

```
for i ← 1 to n − 1 do
    for j ← i + 1 to n do
        if A[i] > A[j] then
            交换 A[i] 和 A[j]
        end
    end
end
```

计算每行伪代码操作次数

算法分析的工具

- **插入排序最坏情况**

```
for j ← 2 to n do ······························ n次
    key ← A[j] ································· n − 1次
    i ← j − 1 ·································· n − 1次
    while i > 0 and A[i] > key do
        A[i + 1] ← A[i]
        i ← i − 1
    end
    A[i + 1] ← key
end
```

- **选择排序最坏情况**

```
for i ← 1 to n − 1 do
    for j ← i + 1 to n do
        if A[i] > A[j] then
            交换 A[i] 和 A[j]
        end
    end
end
```

计算每行伪代码操作次数

算法分析的工具

- **插入排序最坏情况**

```
for j ← 2 to n do ................................ n次
    key ← A[j] ................................ n − 1次
    i ← j − 1 ................................ n − 1次
    while i > 0 and A[i] > key do ··· Σ_{k=2}^{n} k次
        A[i + 1]← A[i] ........... Σ_{k=2}^{n} k − 1次
        i ← i − 1 .................. Σ_{k=2}^{n} k − 1次
    end
    A[i + 1] ← key
end
```

- **选择排序最坏情况**

```
for i ← 1 to n − 1 do
    for j ← i + 1 to n do
        if A[i] > A[j] then
            交换 A[i] 和 A[j]
        end
    end
end
```

计算每行伪代码操作次数

算法分析的工具

- 插入排序最坏情况

```
for j ← 2 to n do ................................ n次
    key ← A[j] ................................ n − 1次
    i ← j − 1 ................................ n − 1次
    while i > 0 and A[i] > key do ··· ∑_{k=2}^{n} k次
        A[i + 1] ← A[i] ........... ∑_{k=2}^{n} k − 1次
        i ← i − 1 .................. ∑_{k=2}^{n} k − 1次
    end
    A[i + 1] ← key ........................ n − 1次
end
```

- 选择排序最坏情况

```
for i ← 1 to n − 1 do
    for j ← i + 1 to n do
        if A[i] > A[j] then
            交换 A[i] 和 A[j]
        end
    end
end
```

计算每行伪代码操作次数

算法分析的工具

- **插入排序最坏情况**

```
for j ← 2 to n do ································ n次
    key ← A[j] ······························ n − 1次
    i ← j − 1 ······························· n − 1次
    while i > 0 and A[i] > key do ··· Σ_{k=2}^{n} k次
        A[i + 1] ← A[i] ············ Σ_{k=2}^{n} k − 1次
        i ← i − 1 ·················· Σ_{k=2}^{n} k − 1次
    end
    A[i + 1] ← key ························· n − 1次
end
```

- **选择排序最坏情况**

```
for i ← 1 to n − 1 do ·························· n次
    for j ← i + 1 to n do ···· Σ_{k=0}^{n−2}(n − k)次
        if A[i] > A[j] then ·· Σ_{k=1}^{n−1}(n − k)次
            交换 A[i] 和 A[j] ········ Σ_{k=1}^{n−1} k次
        end
    end
end
```

计算每行伪代码操作次数

算法分析的工具

- **插入排序最坏情况**

```
for j ← 2 to n do ................................ n次
    key ← A[j] ................................... n − 1次
    i ← j − 1 .................................... n − 1次
    while i > 0 and A[i] > key do ................ Σ_{k=2}^{n} k次
        A[i + 1] ← A[i] .......................... Σ_{k=2}^{n} k − 1次
        i ← i − 1 ................................ Σ_{k=2}^{n} k − 1次
    end
    A[i + 1] ← key ............................... n − 1次
end
```

求和

$$T(n) = \frac{3}{2}n^2 + \frac{7}{2}n - 4$$

- **选择排序最坏情况**

```
for i ← 1 to n − 1 do ............................ n次
    for j ← i + 1 to n do ........................ Σ_{k=0}^{n−2}(n − k)次
        if A[i] > A[j] then ...................... Σ_{k=1}^{n−1}(n − k)次
            交换 A[i] 和 A[j] ..................... Σ_{k=1}^{n−1} k次
        end
    end
end
```

求和

$$T(n) = \frac{3}{2}n^2 + \frac{1}{2}n - 1$$

算法分析的工具

- **插入排序最坏情况**

```
for j ← 2 to n do ........................ n次
    key ← A[j] ........................... n − 1次
    i ← j − 1 ............................ n − 1次
    while i > 0 and A[i] > key do ...... ∑_{k=2}^{n} k次
        A[i + 1] ← A[i] ................ ∑_{k=2}^{n} k − 1次
        i ← i − 1 ...................... ∑_{k=2}^{n} k − 1次
    end
    A[i + 1] ← key ....................... n − 1次
end
```

求和

$$T(n) = \frac{3}{2}n^2 + \frac{7}{2}n - 4$$

- **选择排序最坏情况**

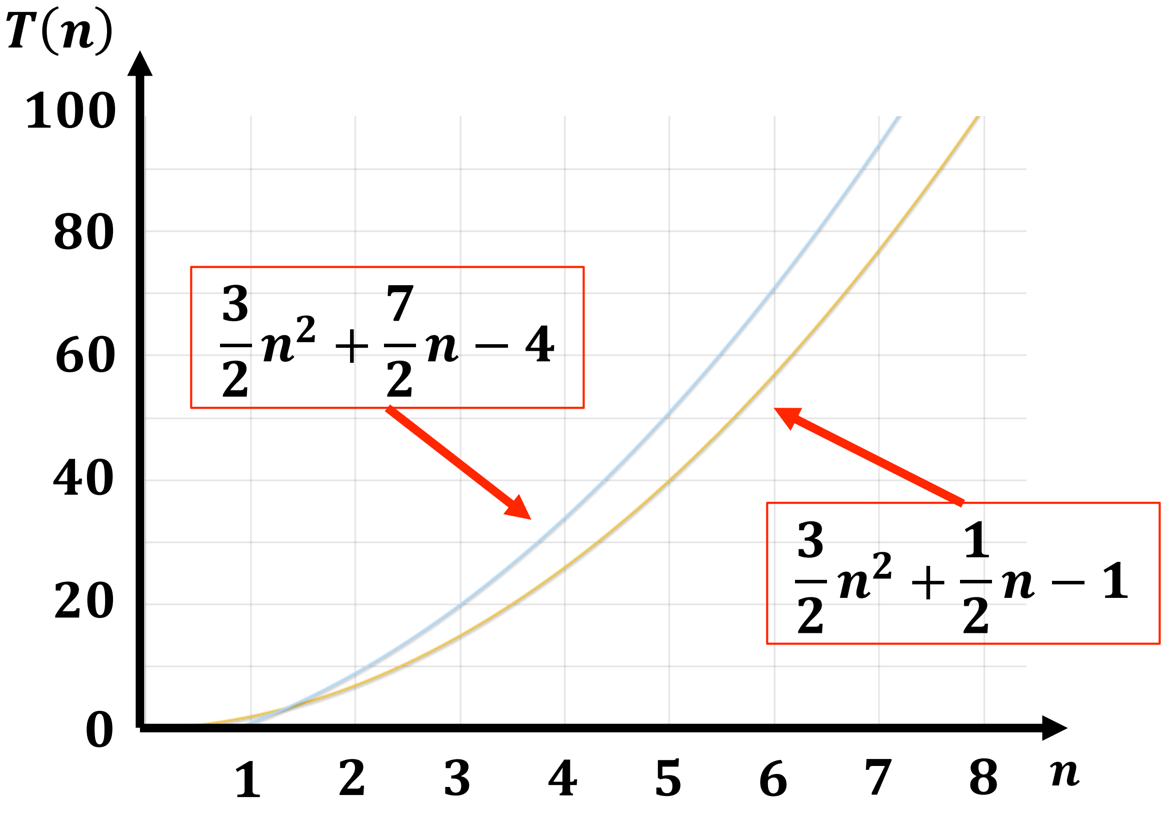
```
for i ← 1 to n − 1 do ..................... n次
    for j ← i + 1 to n do ............... ∑_{k=0}^{n-2}(n − k)次
        if A[i] > A[j] then ............. ∑_{k=1}^{n-1}(n − k)次
            交换 A[i] 和 A[j] ........... ∑_{k=1}^{n-1} k次
        end
    end
end
```

求和

$$T(n) = \frac{3}{2}n^2 + \frac{1}{2}n - 1$$

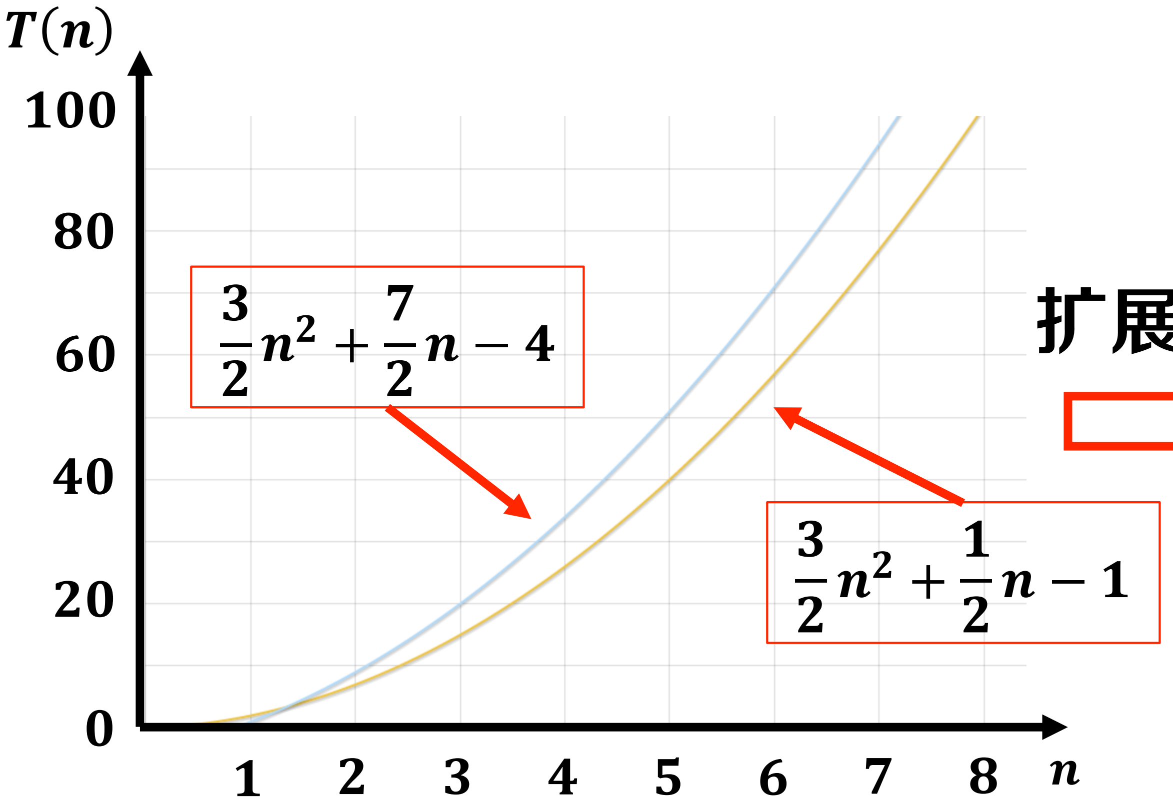
问题：能否简洁地衡量算法运行时间?

算法分析的工具

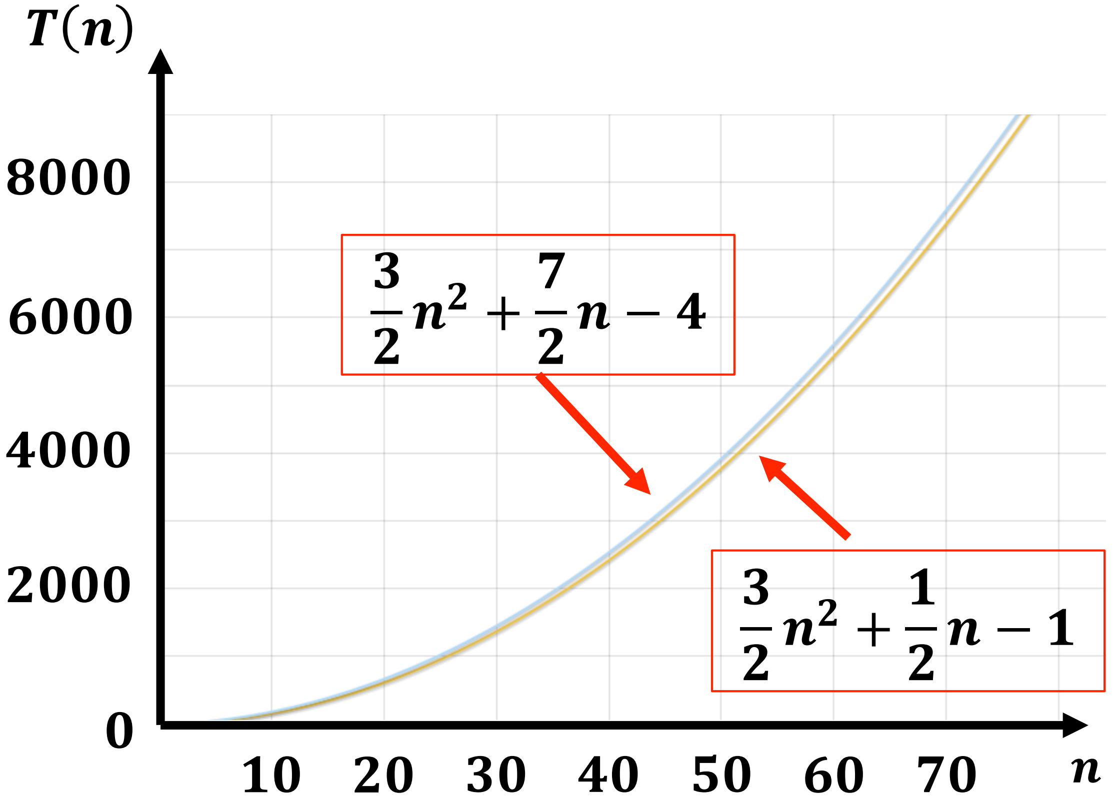


$T(n)$
100
80
60
40
20
0
1
2
3
4
5
6
7
8
n
$\frac{3}{2}n^2+\frac{7}{2}n-4$
$\frac{3}{2}n^2+\frac{1}{2}n-1$

算法分析的工具

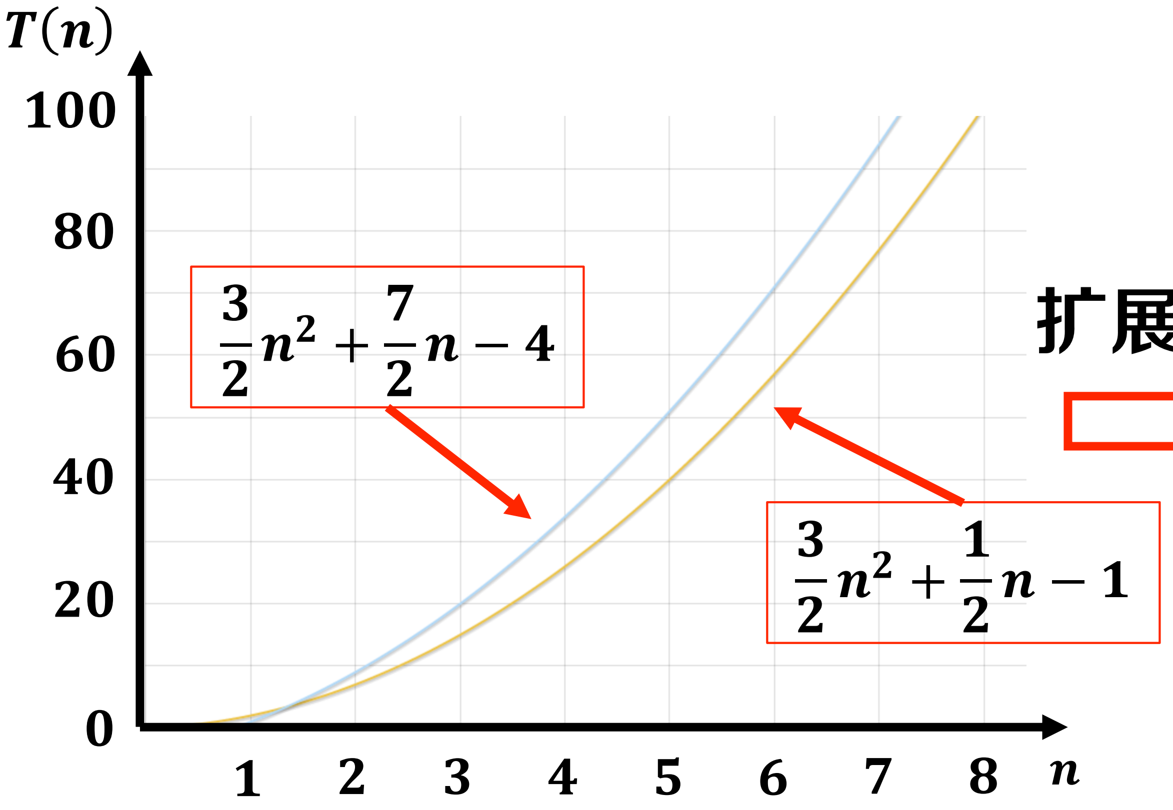


扩展规模

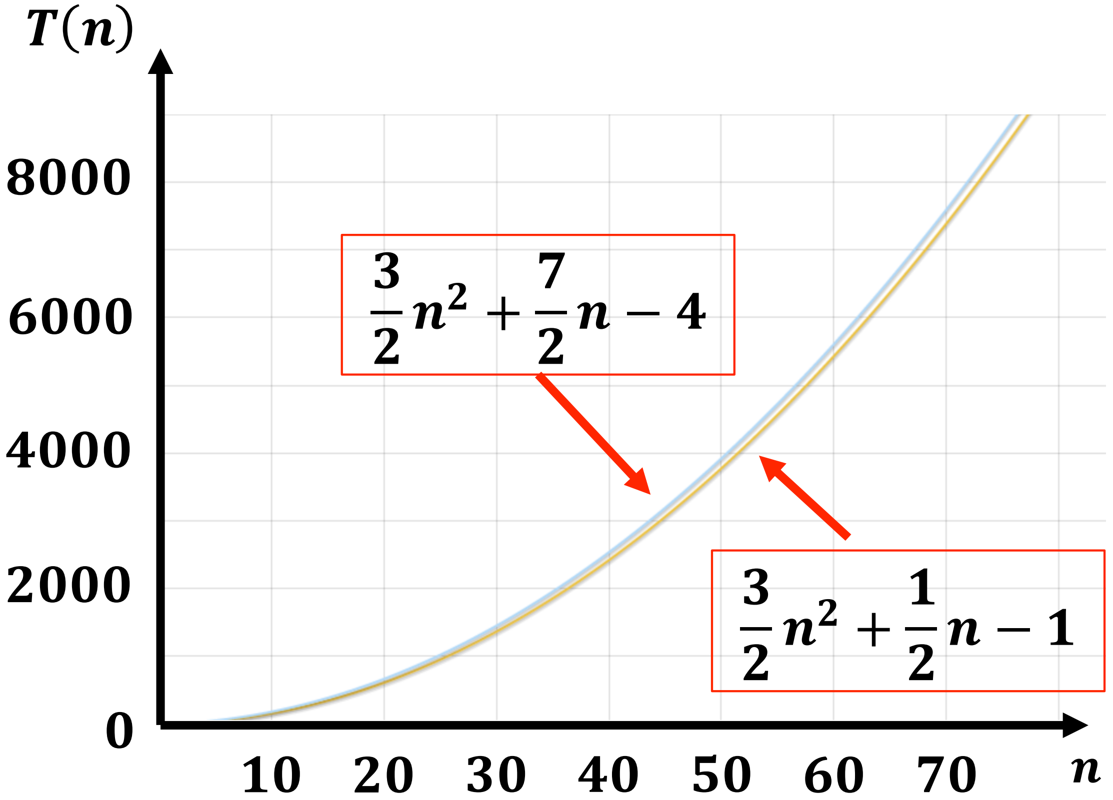


- 在n充分大时，两者相差不大

算法分析的工具

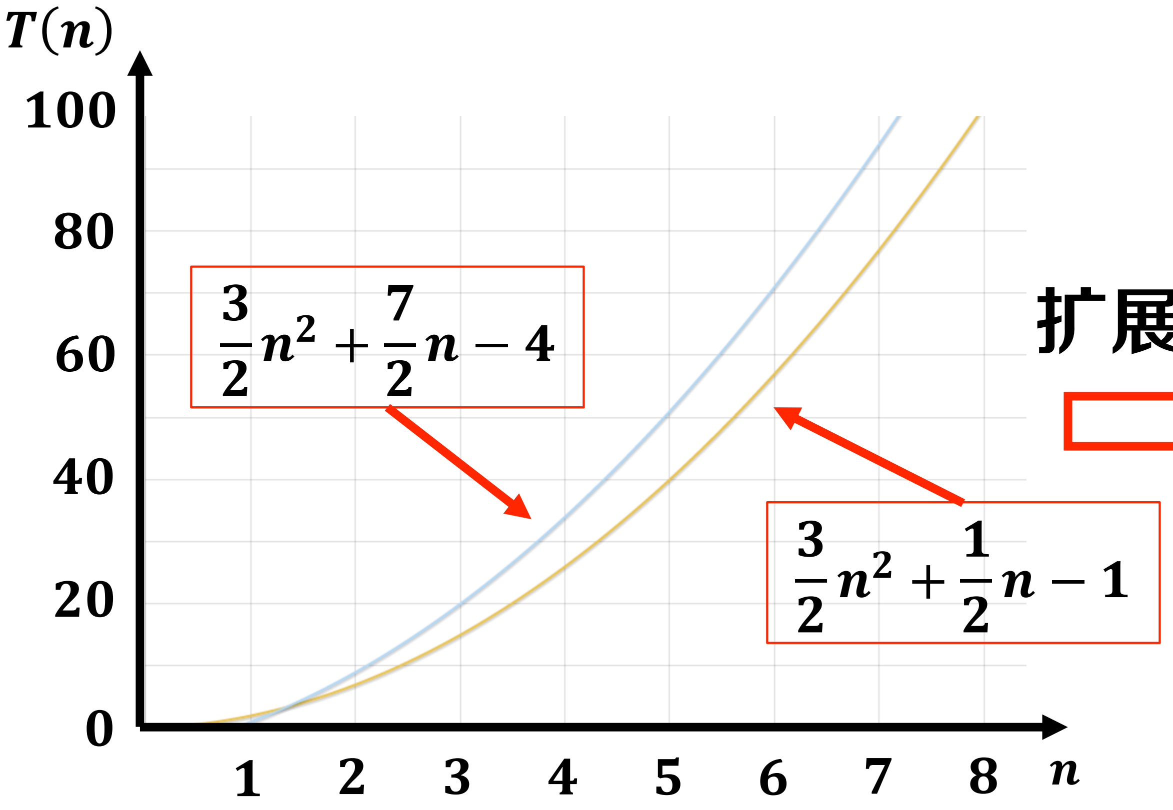


扩展规模

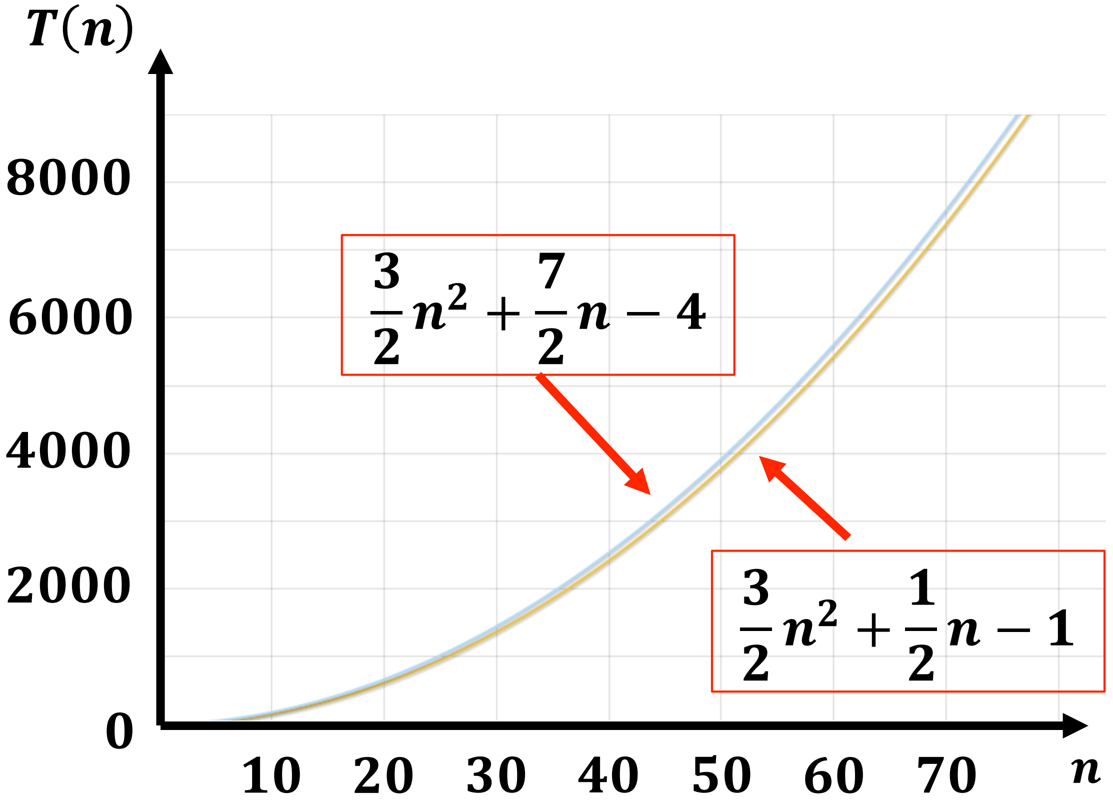


- 在n充分大时，两者相差不大
- 原因?

算法分析的工具



扩展规模



- 在n充分大时，两者相差不大
- 原因：两函数的最高阶项相同

算法分析的工具

- **插入排序最坏情况**

```
for j ← 2 to n do
    key ← A[j]
    i ← j − 1
    while i > 0 and A[i] > key do
        A[i + 1] ← A[i]
        i ← i − 1
    end
    A[i + 1] ← key
end
```

- **选择排序最坏情况**

```
for i ← 1 to n − 1 do
    for j ← i + 1 to n do
        if A[i] > A[j] then
            交换 A[i] 和 A[j]
        end
    end
end
```

$$T(n) = \frac{3}{2}n^2 + \frac{7}{2}n - 4$$

$$T(n) = \frac{3}{2}n^2 + \frac{1}{2}n - 1$$

只保留高阶项.

渐近分析：忽略$T(n)$的系数与低阶项，仅关注高阶项，用记号Θ表示

算法分析的工具

- **插入排序最坏情况**

```
for j ← 2 to n do
    key ← A[j]
    i ← j − 1
    while i > 0 and A[i] > key do
        A[i + 1]← A[i]
        i ← i − 1
    end
    A[i + 1] ← key
end
```

$$T(n) = \Theta(n^2)$$

- **选择排序最坏情况**

```
for i ← 1 to n − 1 do
    for j ← i + 1 to n do
        if A[i] > A[j] then
            交换 A[i] 和 A[j]
        end
    end
end
```

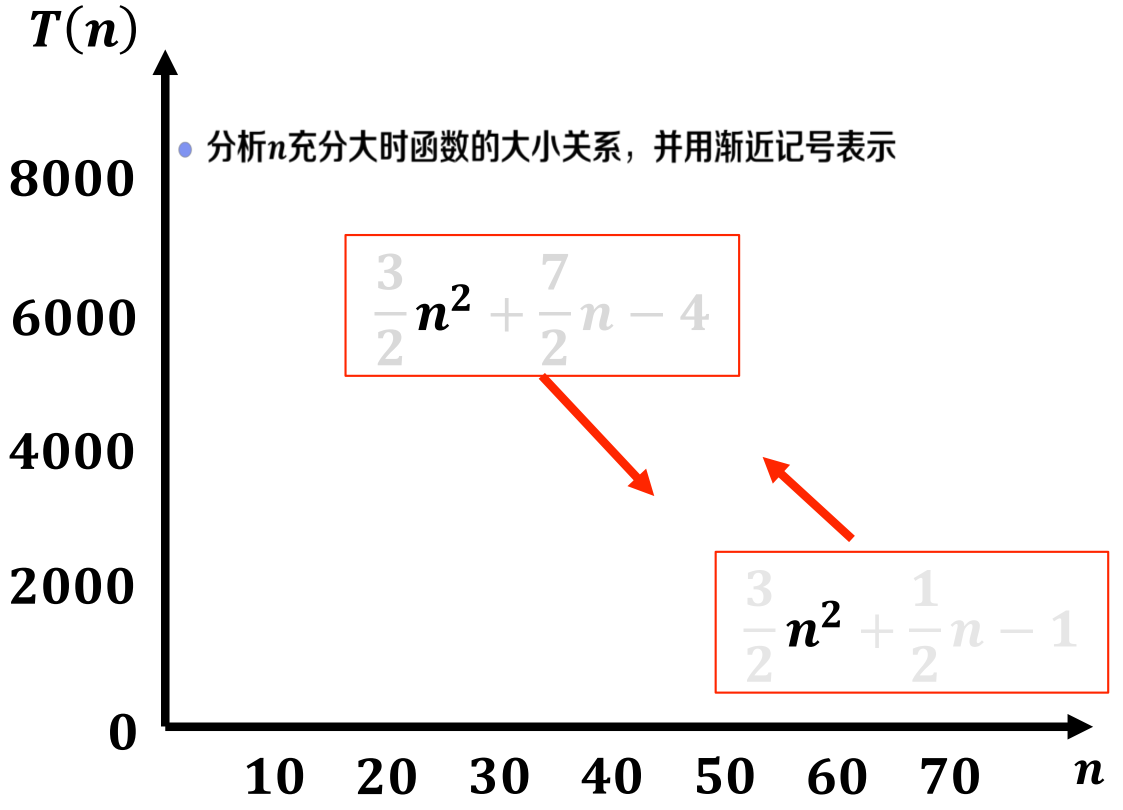
$$T(n) = \Theta(n^2)$$

渐近时间复杂度

渐近分析：忽略$T(n)$的系数与低阶项，仅关注高阶项，用记号Θ表示

渐近分析

- 分析n充分大时函数的大小关系，并用渐近记号表示



渐近记号	名称
$T(n)=\Theta(g(n))$	渐近紧确界 θ
$T(n)=O(g(n))$	渐近上界 O
$T(n)=\Omega(g(n))$	渐近下界 Ω

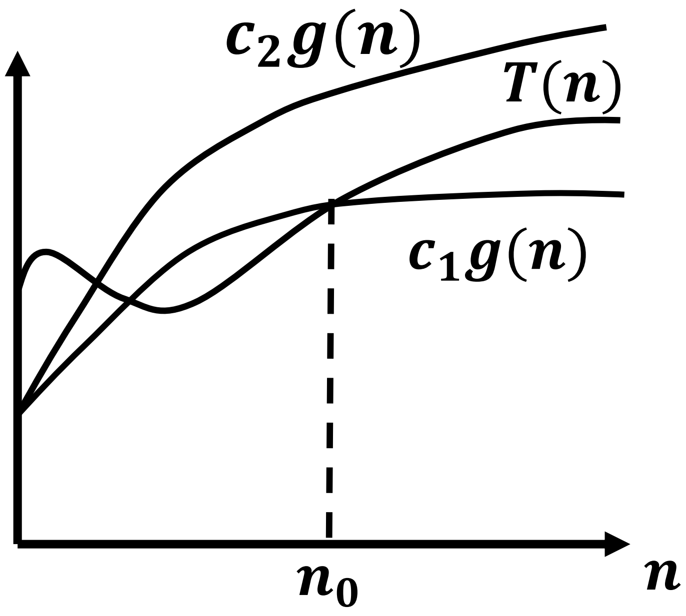
渐近分析：渐近紧确界

Θ记号

定义：

- 对于给定的函数$g(n)$，$\Theta(g(n))$表示以下函数的集合：

$$\Theta(g(n)) = \{T(n): \exists\ c_1, c_2, n_0 > 0, \text{使得} \forall\ n \ge n_0, c_1 g(n) \le T(n) \le c_2 g(n)\}$$



$T(n) = \Theta(g(n))$

渐近分析：渐近紧确界

Θ记号

定义：

- 对于给定的函数$g(n)$，$\Theta(g(n))$表示以下函数的集合：

  $\Theta(g(n)) = \{T(n): \exists\ c_1, c_2, n_0 > 0,$ 使得$\forall\ n \ge n_0, c_1 g(n) \le T(n) \le c_2 g(n)\}$

- **Θ记号示例**
  - $T(n) = \frac{3}{2}n^2 + \frac{7}{2}n - 4 = ?$

渐近分析：渐近紧确界

Θ记号

定义：

- 对于给定的函数$g(n)$，$\Theta(g(n))$表示以下函数的集合：

$$\Theta(g(n)) = \{T(n): \exists\, c_1, c_2, n_0 > 0, \text{使得} \forall\, n \ge n_0, c_1 g(n) \le T(n) \le c_2 g(n)\}$$

- **Θ记号示例**
  - $T(n) = \frac{3}{2}n^2 + \frac{7}{2}n - 4 = ?$
  - 令$n_0 = 2$，当$n \ge n_0$时，有
  - $\frac{3}{2}n^2 + \frac{7}{2}n - 4 \ge \frac{3}{2}n^2 \ge n^2$

c_1, c_2, n_0 代入. 显然存在. 凑定义来证明.

渐近分析：渐近紧确界

Θ记号

定义：

- 对于给定的函数$g(n)$，$\Theta(g(n))$表示以下函数的集合：

$$\Theta\big(g(n)\big) = \{T(n): \exists\, c_1, c_2, n_0 > 0, \text{使得} \forall\, n \geq n_0, c_1 g(n) \leq T(n) \leq c_2 g(n)\}$$

- **Θ记号示例**
  - $T(n) = \frac{3}{2}n^2 + \frac{7}{2}n - 4 = ?$
  - 令$n_0 = 2$，当$n \geq n_0$时，有
  - $\frac{3}{2}n^2 + \frac{7}{2}n - 4 \geq \frac{3}{2}n^2 \geq n^2$
  - $\frac{3}{2}n^2 + \frac{7}{2}n - 4 \leq \frac{3}{2}n^2 + \frac{7}{2}n^2 + n^2 = 6n^2$

渐近分析：渐近紧确界

Θ记号

定义：

- 对于给定的函数$g(n)$，$\Theta(g(n))$表示以下函数的集合：

$$\Theta\big(g(n)\big) = \{T(n): \exists\, c_1, c_2, n_0 > 0, \text{使得} \forall\, n \geq n_0, c_1 g(n) \leq T(n) \leq c_2 g(n)\}$$

Θ记号示例

- $T(n) = \frac{3}{2}n^2 + \frac{7}{2}n - 4 = \Theta(n^2)$
- 令$n_0 = 2$，当$n \geq n_0$时，有
- $\frac{3}{2}n^2 + \frac{7}{2}n - 4 \geq \frac{3}{2}n^2 \geq n^2$
- $\frac{3}{2}n^2 + \frac{7}{2}n - 4 \leq \frac{3}{2}n^2 + \frac{7}{2}n^2 + n^2 = 6n^2$
- 故存在$c_1 = 1, c_2 = 6, n_0 = 2$,使得$\forall\, n \geq n_0, c_1 n^2 \leq T(n) \leq c_2 n^2$

渐近分析：渐近紧确界

- **Θ记号示例**
  - $\frac{3}{2}n^5 + \frac{7}{2}n - 10 =$

渐近分析：渐近紧确界

- **Θ记号示例**
  - $\frac{3}{2}n^5 + \frac{7}{2}n - 10 = \Theta(n^5)$

渐近分析：渐近紧确界

- **Θ记号示例**
  - $\frac{3}{2}n^5 + \frac{7}{2}n - 10 = \Theta(n^5)$
  - $n^3 - n^2 + n =$

渐近分析：渐近紧确界

- $\boldsymbol{\Theta}$记号示例
  - $\frac{3}{2}n^5 + \frac{7}{2}n - 10 = \Theta(n^5)$
  - $n^3 - n^2 + n = \Theta(n^3)$

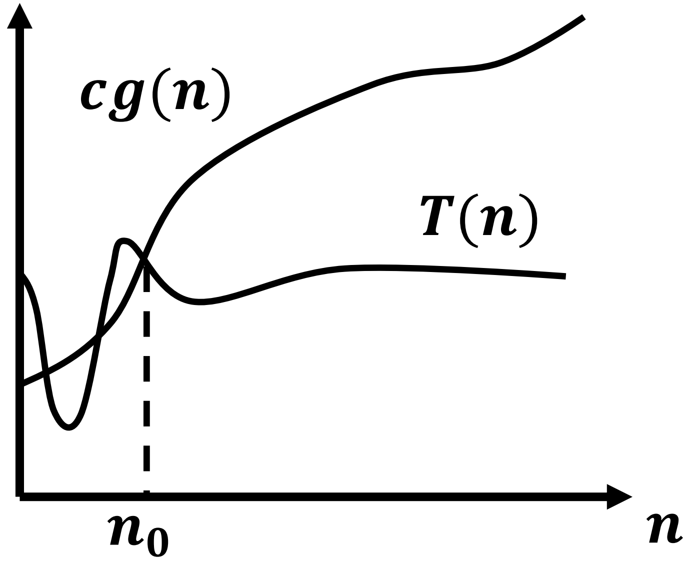
渐近分析：渐近上界

O记号

定义：

- 对于给定的函数$g(n)$，$O(g(n))$表示以下函数的集合：

$$O(g(n)) = \{T(n): \exists\, c, n_0 > 0, \text{使得}\forall\, n \geq n_0, 0 \leq T(n) \leq cg(n)\}$$



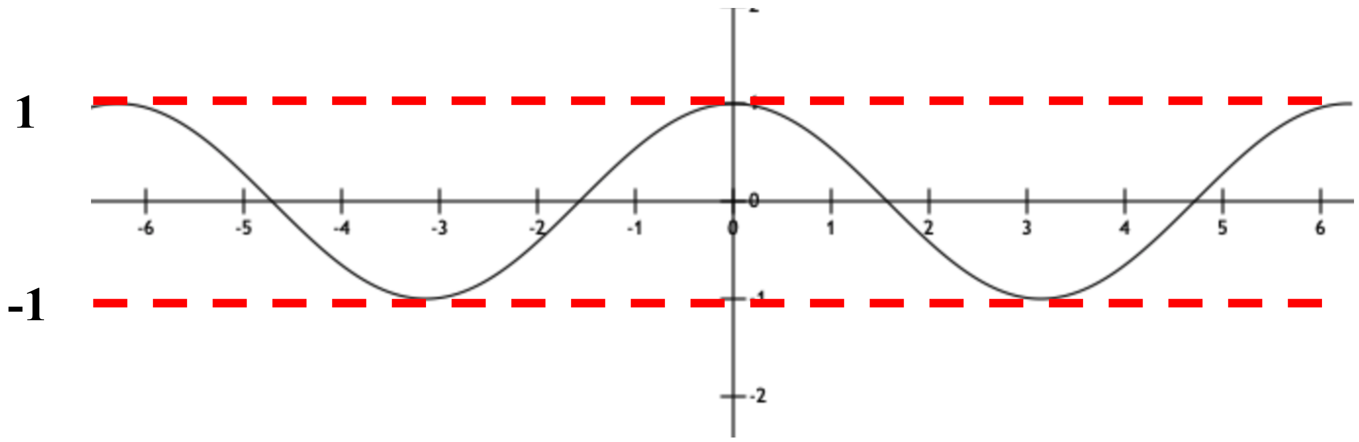
$$T(n) = O(g(n))$$

渐近分析：渐近上界

- O记号示例
  - $\cos n$

渐近分析：渐近上界

- O记号示例
  - $\cos n \le 1$



渐近分析：渐近上界

- $\boldsymbol{O}$记号示例
  - $\cos n = \boldsymbol{O}(1)$

渐近分析：渐近上界

- O记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n =$

渐近分析：渐近上界

- $\boldsymbol{O}$记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O(n^2)$

渐近分析：渐近上界

- $\boldsymbol{O}$记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O(n^2)$
  - $\log_7 n =$

渐近分析：渐近上界

- O记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O(n^2)$
  - $\log_7 n = \frac{\log_2 n}{\log_2 7} =$

对数换底公式

$$\log_x N = \frac{\log_y N}{\log_y x}$$

渐近分析：渐近上界

- O记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O(n^2)$
  - $\log_7 n = \frac{\log_2 n}{\log_2 7} = O(\log_2 n) = O(\log n)$

对数换底公式

$$\log_x N = \frac{\log_y N}{\log_y x}$$

渐近分析：渐近上界

- O记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O(n^2)$
  - $\log_7 n = \frac{\log_2 n}{\log_2 7} = O(\log_2 n) = O(\log n)$
  - $\sum_{i=1}^{n} \frac{1}{i}$　（假设n是2的整数幂）

渐近分析：渐近上界

- O记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O\left(n^2\right)$
  - $\log_7 n = \frac{\log_2 n}{\log_2 7} = O(\log_2 n) = O(\log n)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n}$$

渐近分析：渐近上界

- O记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O(n^2)$
  - $\log_7 n = \frac{\log_2 n}{\log_2 7} = O(\log_2 n) = O(\log n)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n}$$

$$< \frac{1}{1} + \frac{1}{2} + \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \cdots + \frac{1}{\frac{n}{2}} + \frac{1}{n}$$

渐近分析：渐近上界

- O记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O(n^2)$
  - $\log_7 n = \frac{\log_2 n}{\log_2 7} = O(\log_2 n) = O(\log n)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n}$$

$$< \frac{1}{1} + \boxed{\frac{1}{2} + \frac{1}{2}} + \boxed{\frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4}} + \boxed{\frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \cdots + \frac{1}{\frac{n}{2}}} + \frac{1}{n}$$

渐近分析：渐近上界

- O记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O\left(n^2\right)$
  - $\log_7 n = \frac{\log_2 n}{\log_2 7} = O(\log_2 n) = O(\log n)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n}$$

$$< \frac{1}{1} + \frac{1}{2} + \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \cdots + \frac{1}{\frac{\mathrm{n}}{2}} + \frac{1}{n}$$

$$= \frac{1}{1} + 2 \cdot \left(\frac{1}{2}\right) + 4 \cdot \left(\frac{1}{4}\right) + 8 \cdot \left(\frac{1}{8}\right) + \cdots + \frac{n}{2}\left(\frac{1}{\frac{n}{2}}\right) + \frac{1}{n}$$

渐近分析：渐近上界

- O记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O(n^2)$
  - $\log_7 n = \frac{\log_2 n}{\log_2 7} = O(\log_2 n) = O(\log n)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n}$$

$$< \frac{1}{1} + \frac{1}{2} + \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \cdots + \frac{1}{\frac{n}{2}} + \frac{1}{n}$$

$$= \underbrace{\frac{1}{1} + 2 \cdot \left(\frac{1}{2}\right) + 4 \cdot \left(\frac{1}{4}\right) + 8 \cdot \left(\frac{1}{8}\right) + \cdots + \frac{n}{2}\left(\frac{1}{\frac{n}{2}}\right)}_{\log n\text{项}} + \frac{1}{n}$$

渐近分析：渐近上界

- O记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O\left(n^2\right)$
  - $\log_7 n = \frac{\log_2 n}{\log_2 7} = O(\log_2 n) = O(\log n)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n}$$

$$< \frac{1}{1} + \frac{1}{2} + \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \cdots + \frac{1}{\frac{n}{2}} + \frac{1}{n}$$

$$= \frac{1}{1} + 2 \cdot \left(\frac{1}{2}\right) + 4 \cdot \left(\frac{1}{4}\right) + 8 \cdot \left(\frac{1}{8}\right) + \cdots + \frac{n}{2}\left(\frac{1}{\frac{n}{2}}\right) + \frac{1}{n} = \frac{1}{n} + \sum_{j=0}^{\log n - 1} 1$$

渐近分析：渐近上界

- O记号示例
  - $\cos n = O(1)$
  - $\frac{n^2}{2} - 12n = O\left(n^2\right)$
  - $\log_7 n = \frac{\log_2 n}{\log_2 7} = O(\log_2 n) = O(\log n)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n}$$

$$< \frac{1}{1} + \frac{1}{2} + \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \cdots + \frac{1}{\frac{n}{2}} + \frac{1}{n}$$

$$= \frac{1}{1} + 2 \cdot \left(\frac{1}{2}\right) + 4 \cdot \left(\frac{1}{4}\right) + 8 \cdot \left(\frac{1}{8}\right) + \cdots + \frac{n}{2}\left(\frac{1}{\frac{n}{2}}\right) + \frac{1}{n} = \frac{1}{n} + \sum_{j=0}^{\log n - 1} 1$$

$$= \log n + \frac{1}{n} = O(\log n)$$

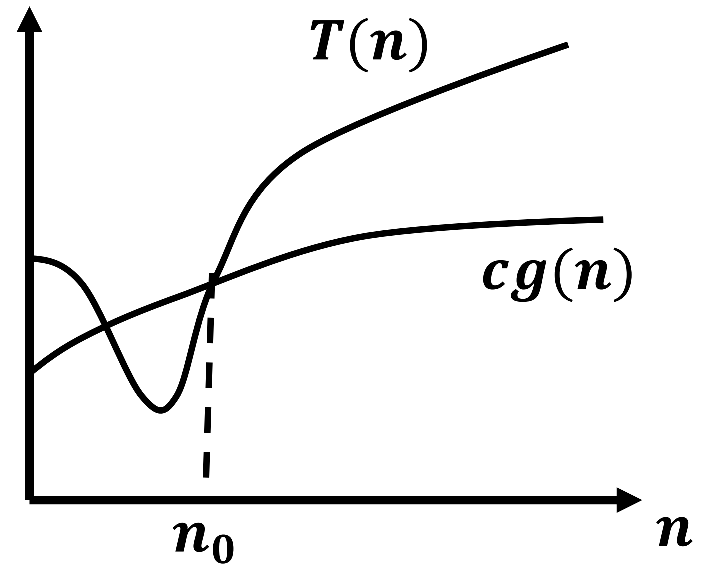
渐近分析：渐近下界

Ω记号

定义：

- 对于给定的函数$g(n)$，$\Omega(g(n))$表示以下函数的集合：

$$\Omega\big(g(n)\big) = \{T(n): \exists\ c, n_0 > 0, \text{使得}\forall\ n \geq n_0, 0 \leq cg(n) \leq T(n)\}$$



$T(n) = \Omega(g(n))$

渐近分析：渐近下界

- Ω记号示例
  - $n^3 - 2n =$

渐近分析：渐近下界

- **Ω记号示例**
  - $n^3 - 2n = \Omega(n^3)$

渐近分析：渐近下界

- **Ω记号示例**
  - $n^3 - 2n = \Omega(n^3)$
  - $n^2 + n = \Omega(n^2)$

渐近分析：渐近下界

- Ω记号示例
  - $n^3 - 2n = \Omega(n^3)$
  - $n^2 + n = \Omega(n^2)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

渐近分析：渐近下界

- **Ω记号示例**
  - $n^3 - 2n = \Omega(n^3)$
  - $n^2 + n = \Omega(n^2)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n}$$

- **Ω记号示例**
  - $n^3 - 2n = \Omega(n^3)$
  - $n^2 + n = \Omega(n^2)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n}$$

$$> \frac{1}{1} + \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \frac{1}{16} + \frac{1}{16} + \cdots \frac{1}{n}$$

渐近分析：渐近下界

- Ω记号示例
  - $n^3 - 2n = \Omega(n^3)$
  - $n^2 + n = \Omega(n^2)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n}$$
$$> \frac{1}{1} + \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \frac{1}{16} + \frac{1}{16} + \cdots \frac{1}{n}$$
$$= \frac{1}{1} + \underbrace{\frac{1}{2} + 2 \cdot \left(\frac{1}{4}\right) + 4 \cdot \left(\frac{1}{8}\right) + 8 \cdot \left(\frac{1}{16}\right) + \cdots + \frac{n}{2}\left(\frac{1}{n}\right)}_{\log n\text{项}}$$

渐近分析：渐近下界

- Ω记号示例
  - $n^3 - 2n = \Omega(n^3)$
  - $n^2 + n = \Omega(n^2)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n}$$
$$> \frac{1}{1} + \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \frac{1}{16} + \frac{1}{16} + \cdots \frac{1}{n}$$
$$= \frac{1}{1} + \frac{1}{2} + 2 \cdot \left(\frac{1}{4}\right) + 4 \cdot \left(\frac{1}{8}\right) + 8 \cdot \left(\frac{1}{16}\right) + \cdots + \frac{n}{2}\left(\frac{1}{n}\right)$$
$$= 1 + \sum_{j=1}^{\log n} \frac{1}{2}$$

渐近分析：渐近下界

- **Ω记号示例**
  - $n^3 - 2n = \Omega(n^3)$
  - $n^2 + n = \Omega(n^2)$
  - $\sum_{i=1}^{n} \frac{1}{i}$ （假设n是2的整数幂）

$$
\begin{aligned}
&= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{n} \\
&> \frac{1}{1} + \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \frac{1}{16} + \frac{1}{16} + \cdots \frac{1}{n} \\
&= \frac{1}{1} + \frac{1}{2} + 2 \cdot \left(\frac{1}{4}\right) + 4 \cdot \left(\frac{1}{8}\right) + 8 \cdot \left(\frac{1}{16}\right) + \cdots + \frac{n}{2}\left(\frac{1}{n}\right) \\
&= 1 + \sum_{j=1}^{\log n} \frac{1}{2} \\
&= 1 + \frac{1}{2}\log n \\
&= \Omega(\log n)
\end{aligned}
$$

渐近分析

- $T(n) = \Theta(g(n))$等价于：$T(n) = \Omega(g(n))$且$T(n) = O(g(n))$

渐近记号	名称
Θ	渐近紧确界
O	渐近上界
Ω	渐近下界

输入情况	情况说明
最好情况	不常出现，不具普遍性
最坏情况	确定上界，更具一般性

算法运行时间称为算法的时间复杂度，通常使用渐近记号O表示

算法分析的实例

- 插入排序最坏情况
- 选择排序最坏情况

$$
\left.
\begin{array}{l}
\textbf{for } j \leftarrow 2 \textit{ to } n \textbf{ do} \\
\quad key \leftarrow A[j] \\
\quad i \leftarrow j-1 \\
\quad \left.\begin{array}{l}
\textbf{while } i > 0 \textit{ and } A[i] > key \textbf{ do} \\
\quad A[i+1] \leftarrow A[i] \\
\quad i \leftarrow i-1 \\
\textbf{end}
\end{array}\right\} O(n) \\
\quad \mathrm{A}[i+1] \leftarrow key \\
\textbf{end}
\end{array}
\right\} O(n^2)
$$

$$
\left.
\begin{array}{l}
\textbf{for } i \leftarrow 1 \textit{ to } n-1 \textbf{ do} \\
\quad \left.\begin{array}{l}
\textbf{for } j \leftarrow i+1 \textit{ to } n \textbf{ do} \\
\quad \textbf{if } A[i] > A[j] \textbf{ then} \\
\quad\quad \textbf{交换 } A[i] \textbf{ 和 } A[j] \\
\quad \textbf{end} \\
\textbf{end}
\end{array}\right\} O(n) \\
\textbf{end}
\end{array}
\right\} O(n^2)
$$

算法的分析小结

- 算法分析的原则

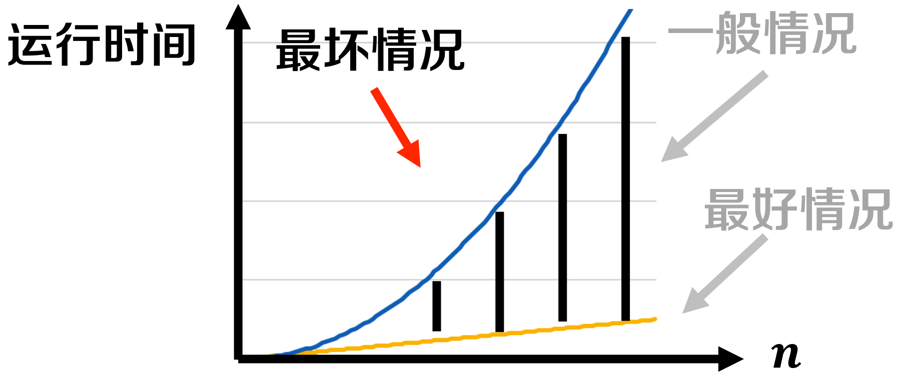
+	−	×	÷
:=	>	<	=

统一机器性能

算法的分析小结

- 算法分析的原则

+	−	×	÷
:=	>	<	=



统一机器性能

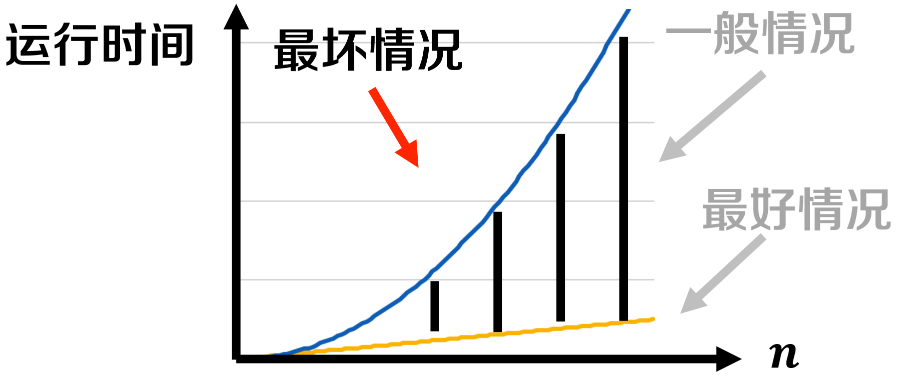
分析最坏情况

算法的分析小结

- 算法分析的原则

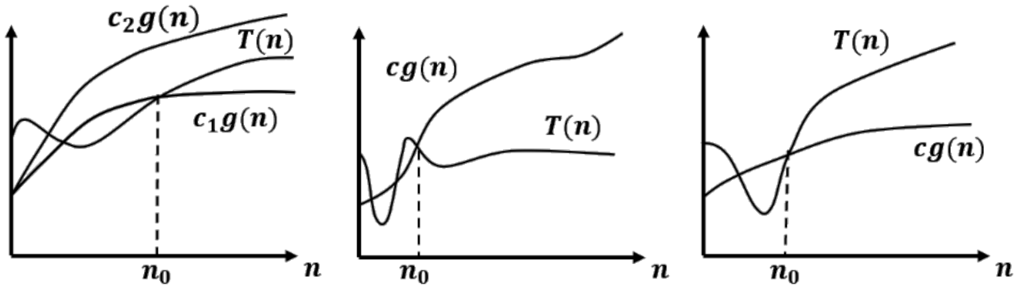
+	−	×	÷
:=	>	<	=

统一机器性能



分析最坏情况

- 算法分析的工具



$T(n) = \Theta(g(n))$ $T(n) = O(g(n))$ $T(n) = \Omega(g(n))$

渐近上界O

采用渐近分析

课程绪论

算法的由来

算法的定义

算法的性质

算法的表示

算法的分析

课程规划

算法设计与分析

分而治之篇

- 归并排序
- 递归式求解
- 最大子数组问题 I
- 逆序对计数问题
- 快速排序
- 次序选择问题

动态规划篇

- 0-1背包问题
- 最大子数组问题 II
- 最长公共子序列问题
- 最长公共子串问题
- 编辑距离问题
- 钢条切割问题
- 矩阵链乘法问题

贪心策略篇

- 部分背包问题
- 霍夫曼编码
- 活动选择问题