Design and Analysis of Algorithms

Part II: Dynamic Programming

Lecture 11: Maximum Contiguous Subarray Problem II

童咏昕

北京航空航天大学
计算机学院

动态规划篇概述

- 在算法课程第二部分动态规划主题中，我们将主要聚焦于如下经典问题：
  - 0-1 Knapsack (0-1背包问题)
  - Maximum Contiguous Subarray II (最大连续子数组 II)
  - Longest Common Subsequences (最长公共子序列)
  - Longest Common Substrings (最长公共子串)
  - Minimum Edit Distance (最小编辑距离)
  - Rod-Cutting (钢条切割)
  - Chain Matrix Multiplication (矩阵链乘法)

动态规划篇概述

- 在算法课程第二部分“动态规划”主题中，我们将主要聚焦于如下经典问题：
  - 0-1 Knapsack (0-1背包问题)
  - Maximum Contiguous Subarray II (最大连续子数组 II)
  - Longest Common Subsequences (最长公共子序列)
  - Longest Common Substrings (最长公共子串)
  - Minimum Edit Distance (最小编辑距离)
  - Rod-Cutting (钢条切割)
  - Chain Matrix Multiplication (矩阵链乘法)

最大子数组问题

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

最大子数组问题

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

- 子数组为数组X中**连续**的一段序列

最大子数组问题

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

- 子数组$X[3..7]$

最大子数组问题

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

- 子数组$X[3..7]$
  - 求和为：$3 + 5 - 4 + 3 + 2 = 9$

最大子数组问题

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

- 子数组$X[3..7]$
  - 求和为：$3+5-4+3+2=9$
- 子数组$X[1..10]$

最大子数组问题

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

- 子数组$X[3..7]$
  - 求和为：$3 + 5 - 4 + 3 + 2 = 9$
- 子数组$X[1..10]$
  - 求和为：$-1 - 3 + 3 + 5 - 4 + 3 + 2 - 2 + 3 + 6 = 12$

最大子数组问题

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

- 子数组$X[3..7]$
  - 求和为：$3+5-4+3+2=9$
- 子数组$X[1..10]$
  - 求和为：$-1-3+3+5-4+3+2-2+3+6=12$
- 子数组$X[3..10]$

最大子数组问题

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

- 子数组$X[3..7]$
  - 求和为：$3+5-4+3+2=9$
- 子数组$X[1..10]$
  - 求和为：$-1-3+3+5-4+3+2-2+3+6=12$
- 子数组$X[3..10]$
  - 求和为：$3+5-4+3+2-2+3+6=16$

最大子数组问题

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

- 子数组$X[3..7]$
  - 求和为：$3+5-4+3+2=9$
- 子数组$X[1..10]$
  - 求和为：$-1-3+3+5-4+3+2-2+3+6=12$
- 子数组$X[3..10]$
  - 求和为：$3+5-4+3+2-2+3+6=16$

问题：寻找数组X最大的非空子数组？

最大子数组问题

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

- 子数组$X[3..7]$
  - 求和为：$3+5-4+3+2=9$
- 子数组$X[1..10]$
  - 求和为：$-1-3+3+5-4+3+2-2+3+6=12$
- 子数组$X[3..10]$
  - 求和为：$3+5-4+3+2-2+3+6=16$

问题：寻找数组X最大的非空子数组?
答案：$X[3..10]=16$

问题定义

- 形式化定义

最大子数组问题

Max Continuous Subarray, MCS

输入

- 给定一个数组$X[1..n]$，对于任意一对数组下标为$l, r\ (l \le r)$的**非空子数组**，其和记为

$$S(l,r) = \sum_{i=l}^{r} X[i]$$

输出

- 求出$S(l,r)$的**最大值，记为S_{max}**

蛮力枚举

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

- 数组$X[1..n]$，其所有的下标$l,r(l \leq r)$组合分为以下两种情况
  - 当$l = r$时，一共$C_n^1 = n$种组合
  - 当$l < r$时，一共C_n^2种组合

- 枚举$n + C_n^2$种下标l,r组合，求出最大子数组之和

蛮力枚举

- $l = 3,\ r = 8$
- 计算$S(3,8)$：

$l = 3$ $\quad$ $r = 8$

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

$$S(3,8) = 3 + 5 - 4 + 3 + 2 - 2 = 7$$

蛮力枚举

- 枚举$n + C_n^2$种可能的区间$[l, r](l \leq r)$，求解最大子数组之和S_{max}

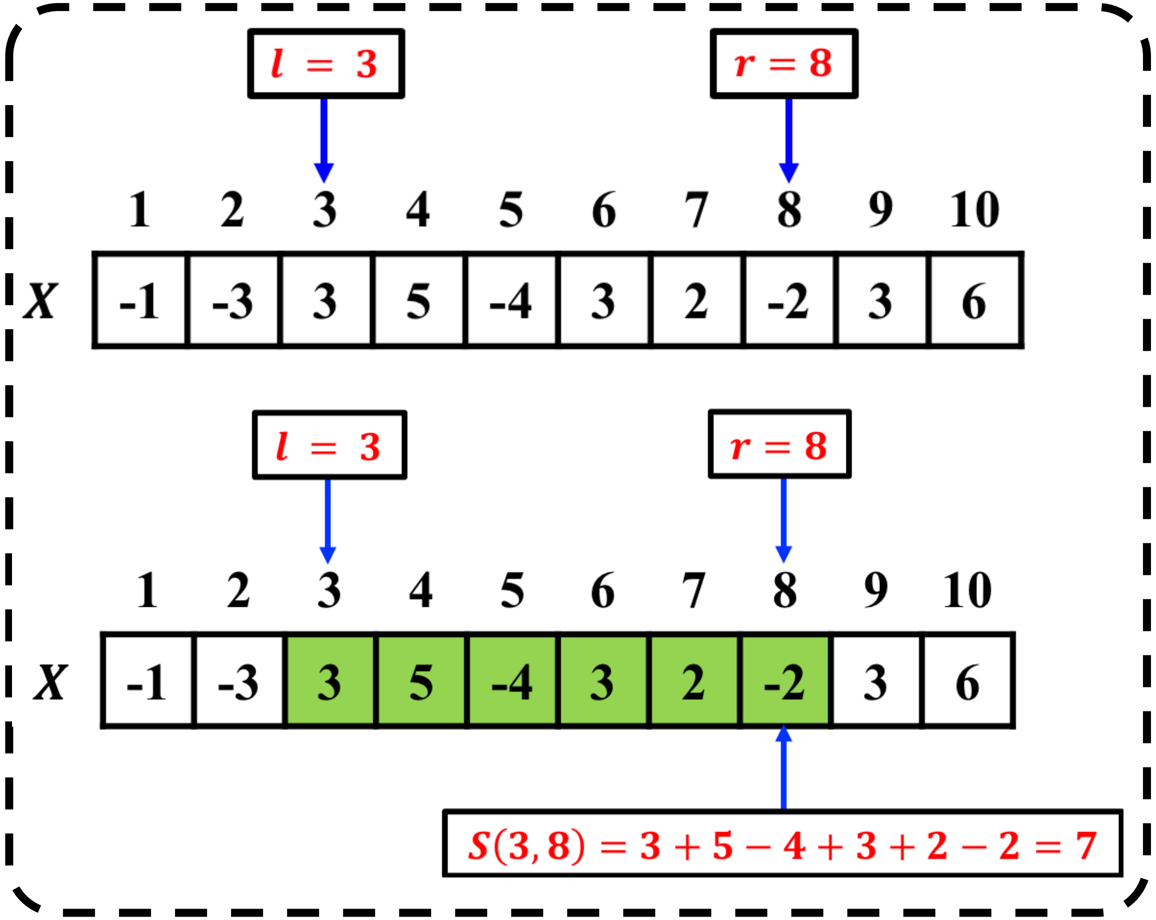
输入: 数组$X[1..n]$
输出: 最大子数组之和S_{max}

```
S_max ← −∞
for l ← 1 to n do
    for r ← l to n do
        S(l,r) ← 0
        for i ← l to r do
            S(l,r) ← S(l,r) + X[i]
        end
        S_max ← max{S_max, S(l,r)}
    end
end
return S_max
```

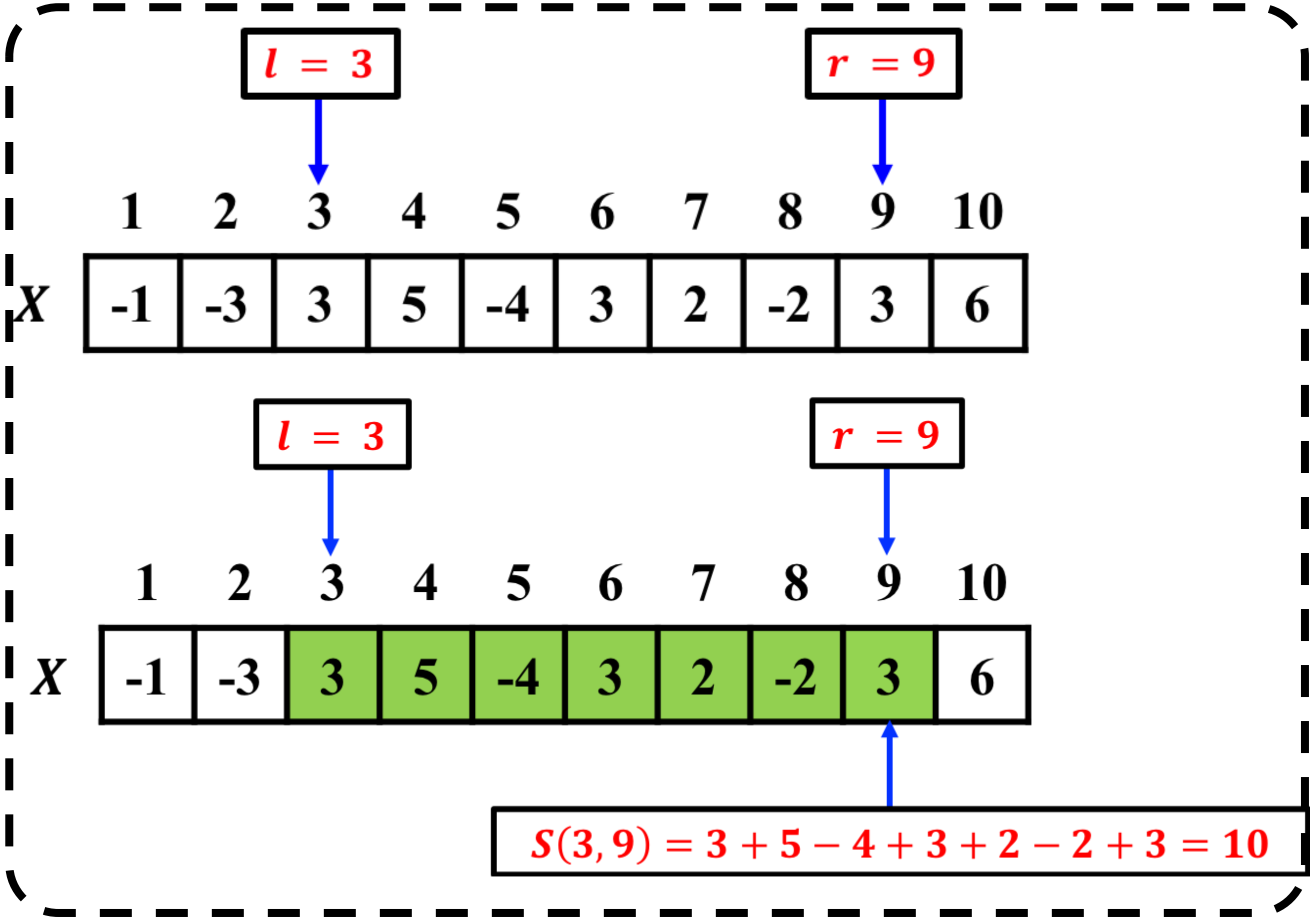
$O(n)$ $O(n^2)$ $O(n^3)$

时间复杂度：$O(n^3)$

从蛮力枚举到优化枚举

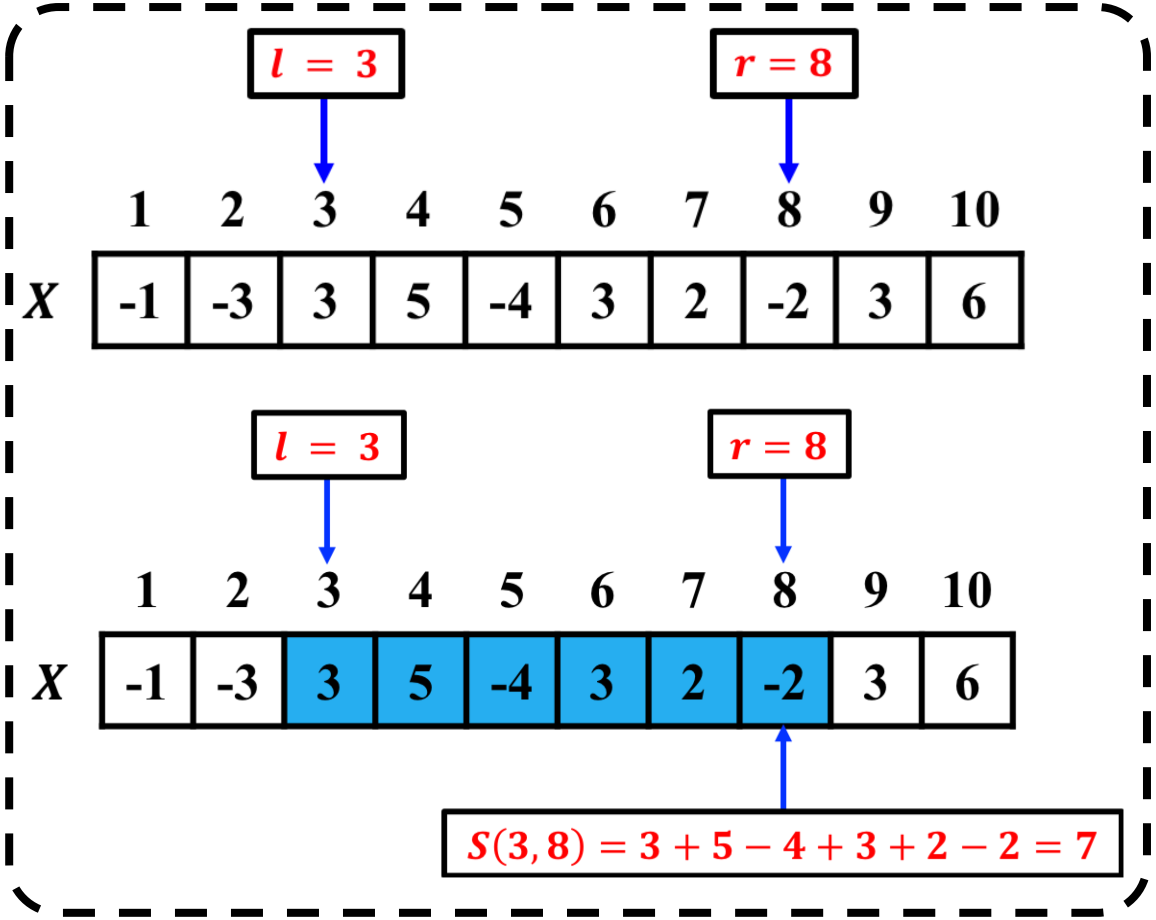


计算$S(3,8)$循环6次

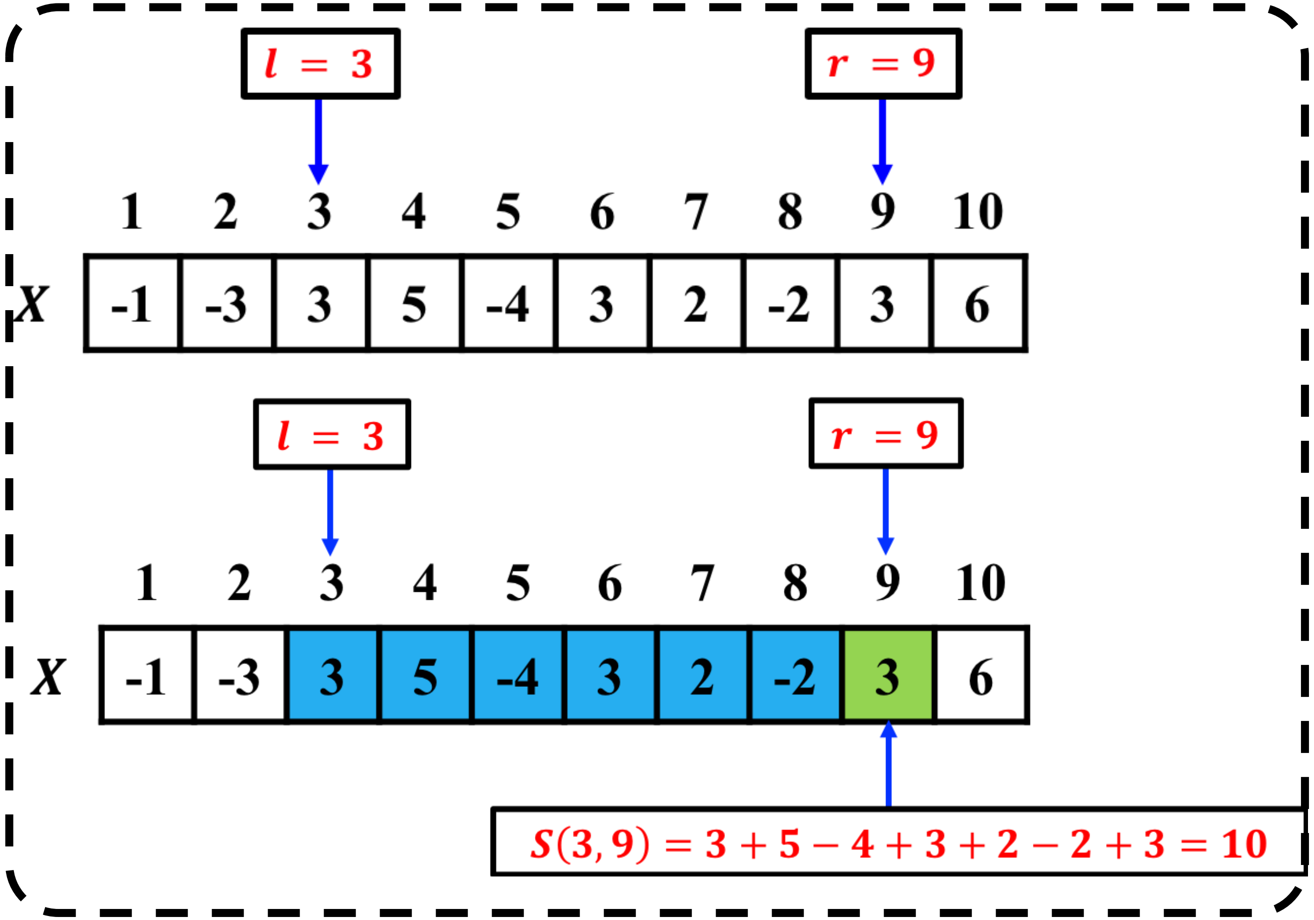


计算$S(3,9)$循环7次

从蛮力枚举到优化枚举



计算$S(3,8)$循环6次



计算$S(3,9)$循环7次

优化枚举

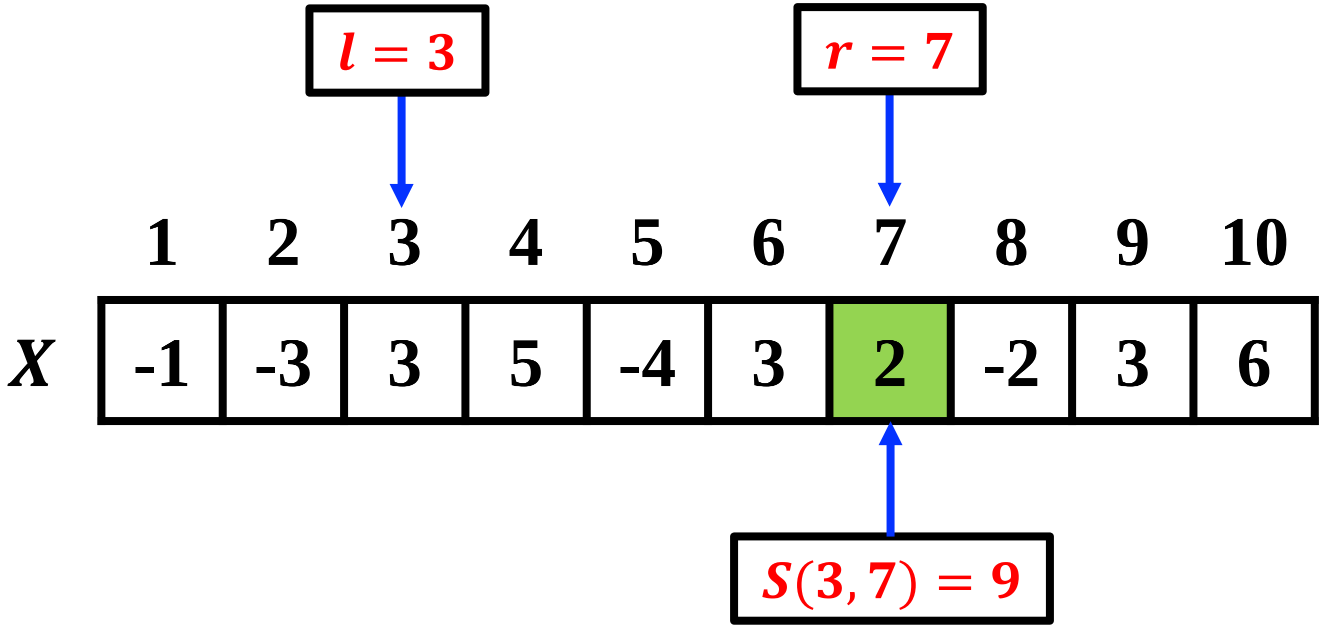
- $l = 3, \ r = 7$

$l = 3$ $\quad$ $r = 7$

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

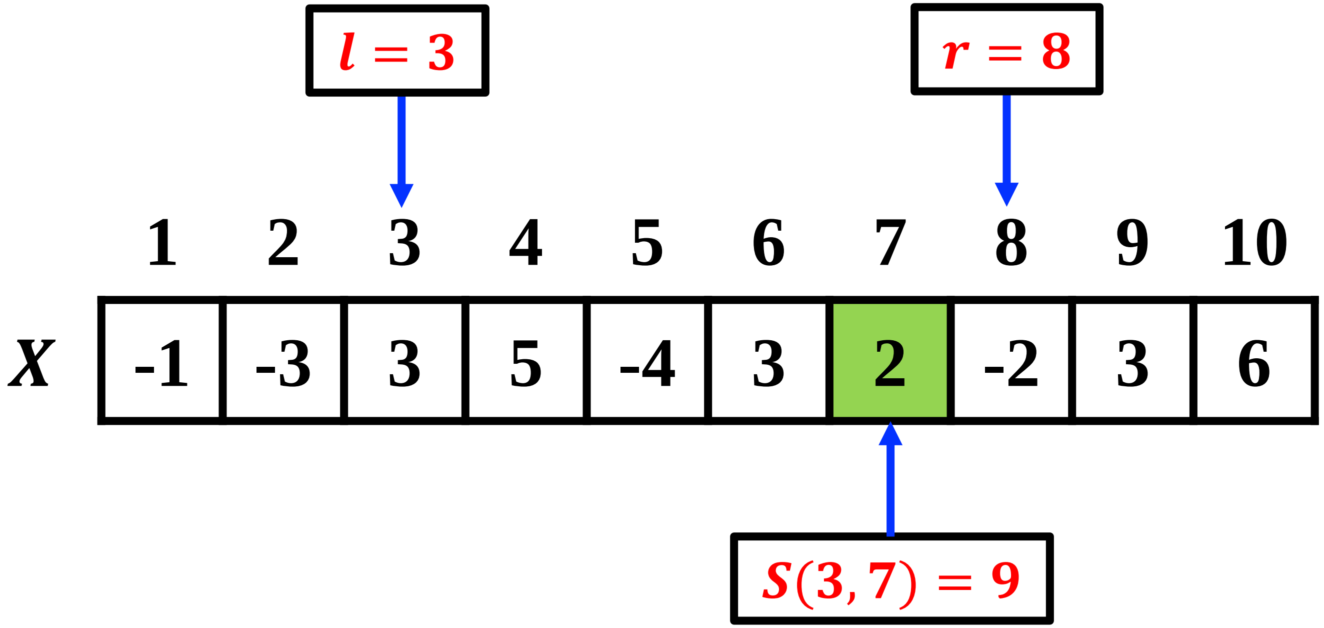
优化枚举

- $l = 3,\ r = 7,\ S(3,7) = 9$



优化枚举

- $l = 3,\ r = 8,\ S(3,8) = ?$



优化枚举

- $l = 3,\ r = 8,\ S(3,8) = 7$

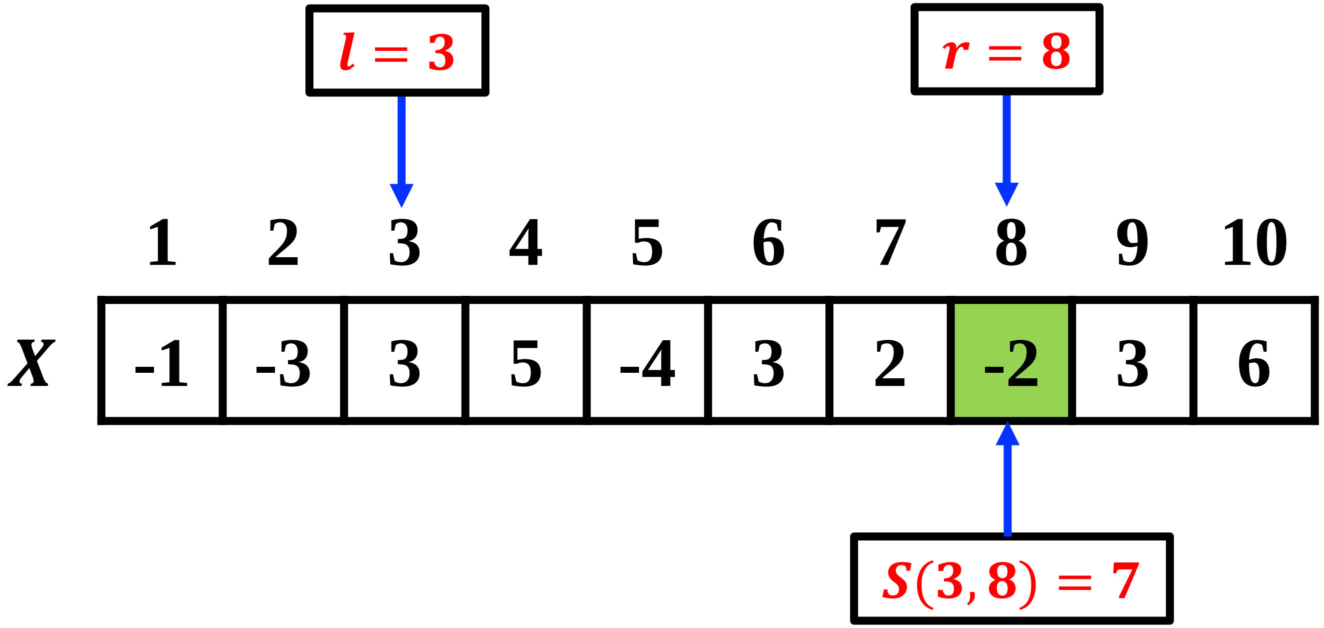
$\boxed{l = 3}$ $\boxed{r = 8}$

	1	2	3	4	5	6	7	8	9	10
X	-1	-3	3	5	-4	3	2	-2	3	6

$\boxed{S(3,8) = S(3,7) - 2 = 7}$

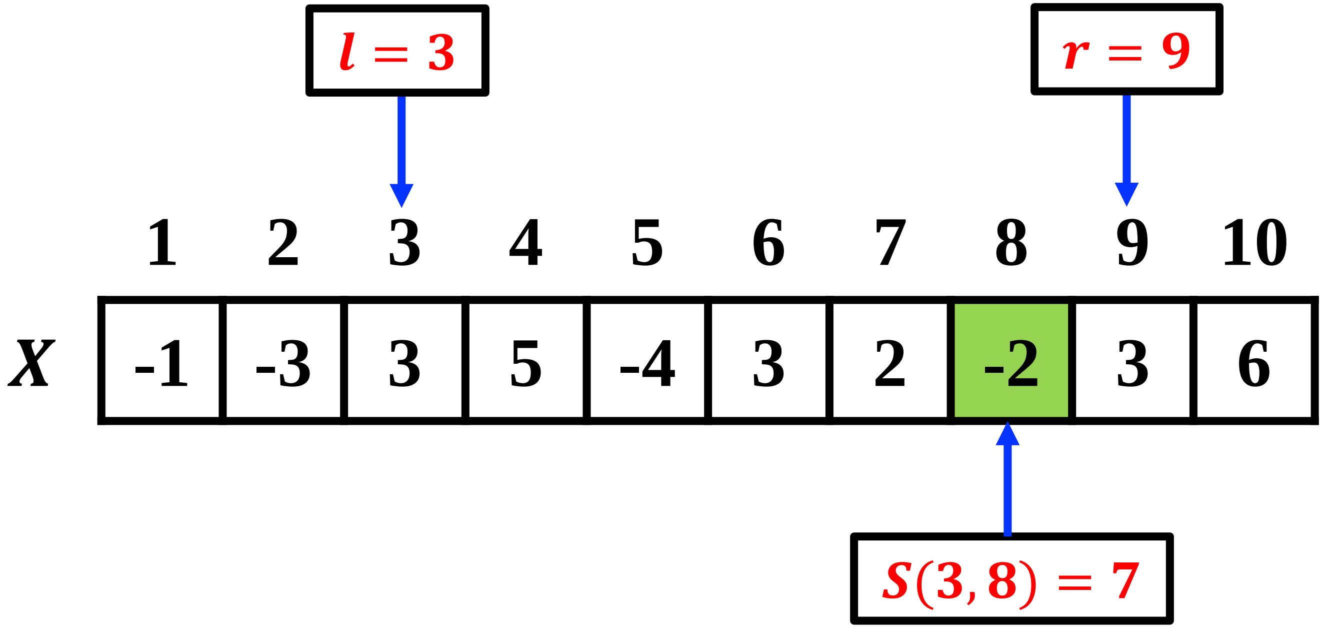
优化枚举

- $l=3,\ r=8,\ S(3,8)=7$



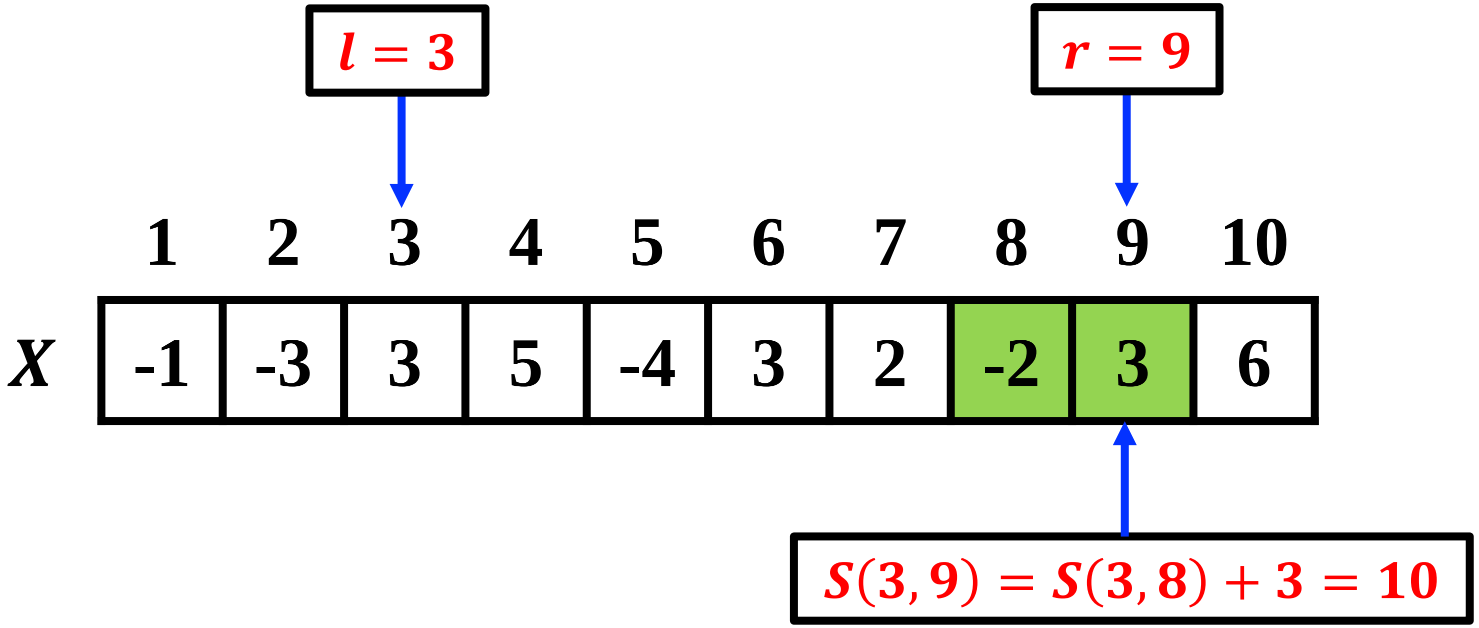
优化枚举

- $l = 3$, $r = 9$, $S(3, 9) = ?$



优化枚举

- $l = 3,\ r = 9,\ S(3,9) = ?$



优化枚举

- 核心思想: $S(l,r)=\sum_{i=l}^{r}X[i]=\ S(l,r-1)+X[r]$

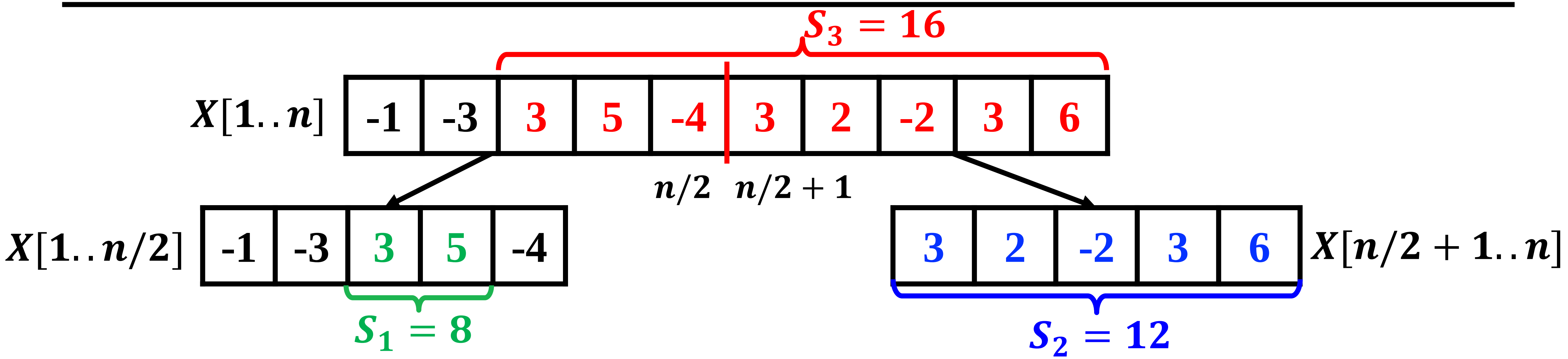
输入: 数组$X[1..n]$
输出: 最大子数组之和S_{max}

$S_{max} \leftarrow -\infty$
for $l \leftarrow 1$ *to* n **do**
  $S \leftarrow 0$
  for $r \leftarrow l$ *to* n **do**
    $S \leftarrow S + X[r]$
    $S_{max} \leftarrow \max\{S_{max}, S\}$
  end
end
return S_{max}

$O(n)$ $O(n^2)$

时间复杂度: $O(n^2)$

分而治之



- 将数组$X[1..n]$分为$X[1..n/2]$和$X[n/2+1..n]$
- 递归求解子问题
  - S_1：数组$X[1..n/2]$的最大子数组
  - S_2：数组$X[n/2+1..n]$的最大子数组
- 合并子问题，得到S_{max}
  - S_3：跨中点的最大子数组
  - 数组X的最大子数组之和$S_{max} = max\{S_1, S_2, S_3\}$

分解原问题

解决子问题

合并问题解

分而治之

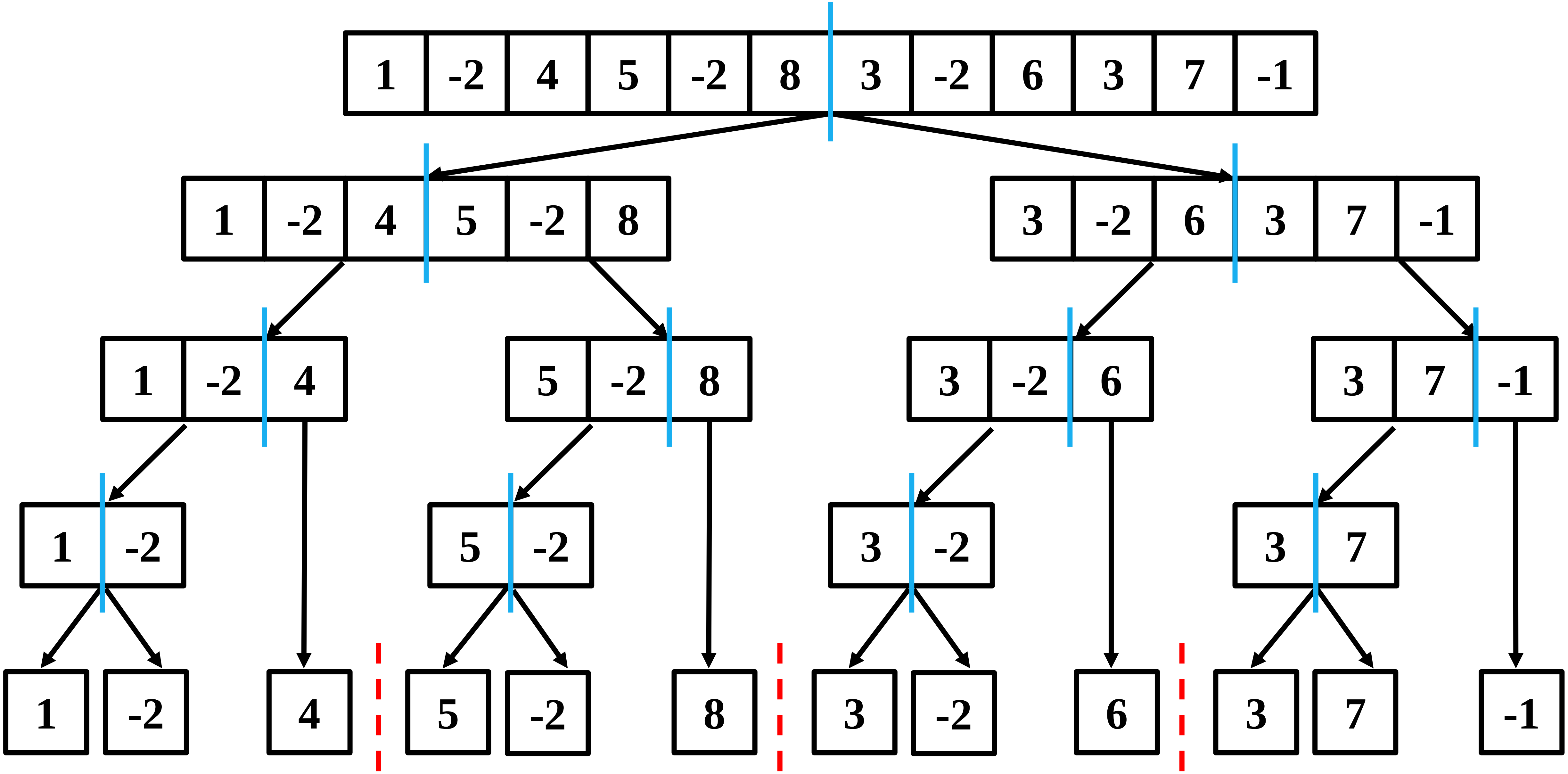
- **时间复杂度分析**

输入: 数组X, 数组下标$low, high$
输出: 最大子数组之和S_{max}

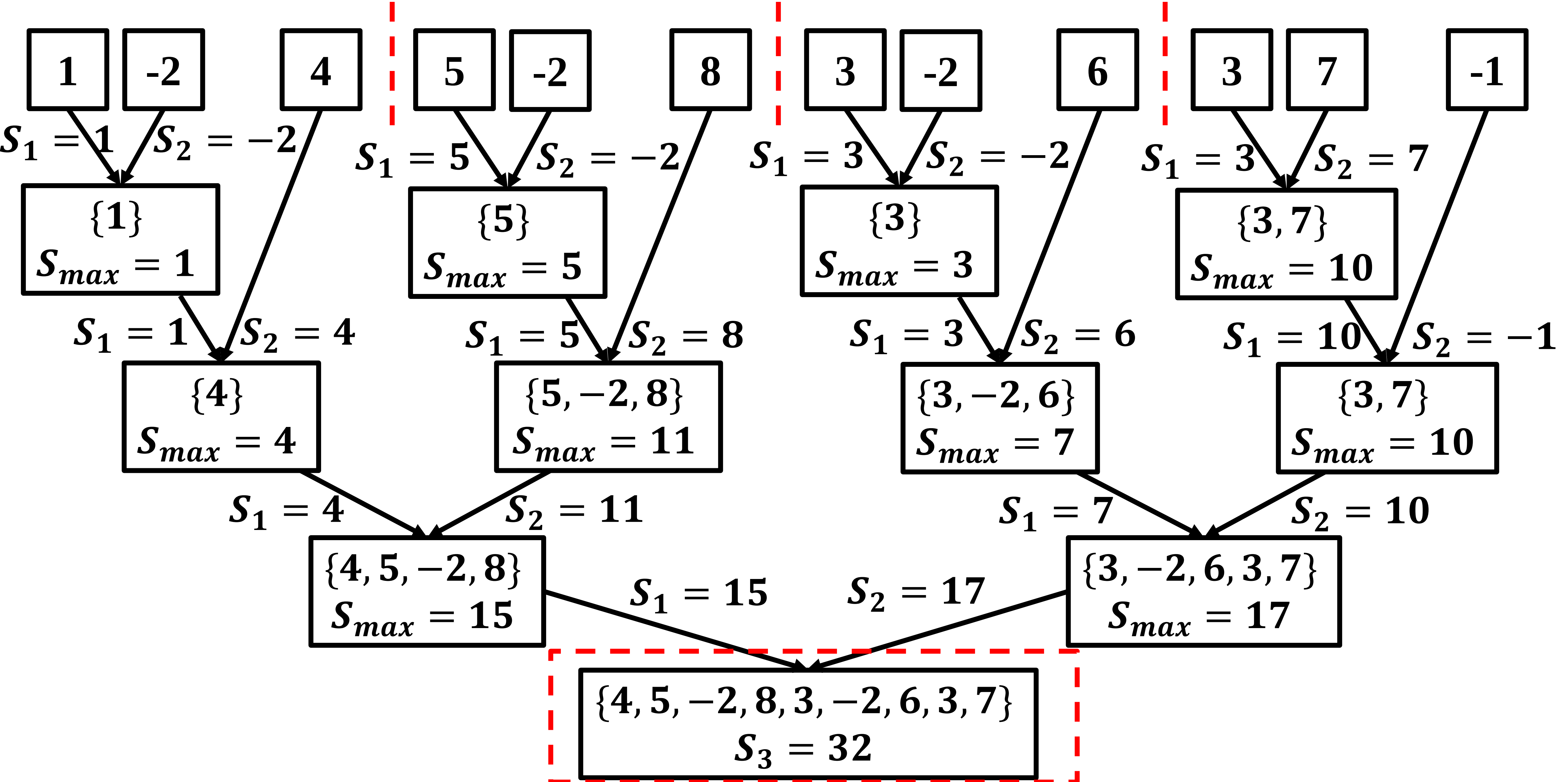
if $low = high$ **then**
  return $X[low]$
end
else
  $mid \leftarrow \lfloor \frac{low+high}{2} \rfloor$
  $S_1 \leftarrow$ MaxSubArray(X,low,mid)
  $S_2 \leftarrow$ MaxSubArray(X,mid+1,high)
  $S_3 \leftarrow$ CrossingSubArray(X,low,mid,high)
  $S_{max} \leftarrow \max\{S_1, S_2, S_3\}$
  return S_{max}
end

时间复杂度：$O(n \log n)$

分而治之


1 -2 4 5 -2 8 3 -2 6 3 7 -1
1 -2 4 5 -2 8
3 -2 6 3 7 -1
1 -2 4
5 -2 8
3 -2 6
3 7 -1
1 -2
5 -2
3 -2
3 7
1
-2
4
5
-2
8
3
-2
6
3
7
-1

算法实例


1
-2
4
5
-2
8
3
-2
6
3
7
-1
$S_1 = 1$
$S_2 = -2$
$S_1 = 5$
$S_2 = -2$
$S_1 = 3$
$S_2 = -2$
$S_1 = 3$
$S_2 = 7$
$\{1\}$
$S_{max} = 1$
$\{5\}$
$S_{max} = 5$
$\{3\}$
$S_{max} = 3$
$\{3, 7\}$
$S_{max} = 10$
$S_1 = 1$
$S_2 = 4$
$S_1 = 5$
$S_2 = 8$
$S_1 = 3$
$S_2 = 6$
$S_1 = 10$
$S_2 = -1$
$\{4\}$
$S_{max} = 4$
$\{5, -2, 8\}$
$S_{max} = 11$
$\{3, -2, 6\}$
$S_{max} = 7$
$\{3, 7\}$
$S_{max} = 10$
$S_1 = 4$
$S_2 = 11$
$S_1 = 7$
$S_2 = 10$
$\{4, 5, -2, 8\}$
$S_{max} = 15$
$S_1 = 15$
$S_2 = 17$
$\{3, -2, 6, 3, 7\}$
$S_{max} = 17$
$\{4, 5, -2, 8, 3, -2, 6, 3, 7\}$
$S_3 = 32$

算法比较

算法名称	时间复杂度
蛮力枚举	$O(n^3)$
优化枚举	$O(n^2)$
分而治之	$O(n\log n)$
？	$O(n)$

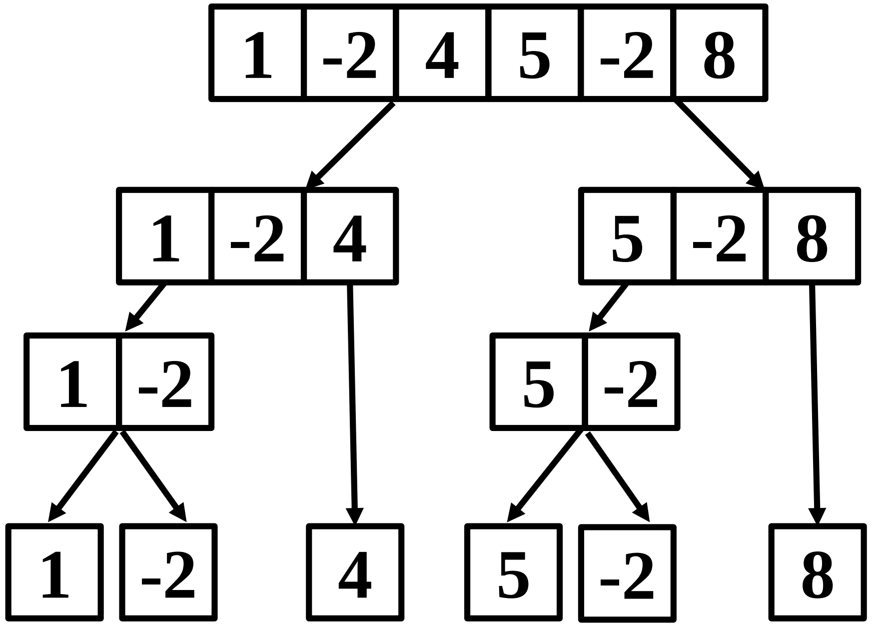
问题：是否可以设计一个时间复杂度为$O(n)$的算法?

算法比较

算法名称	时间复杂度
蛮力枚举	$O(n^3)$
优化枚举	$O(n^2)$
分而治之	$O(n\log n)$
动态规划	$O(n)$

算法比较

算法名称	时间复杂度
蛮力枚举	$O(n^3)$
优化枚举	$O(n^2)$
分而治之	$O(n\log n)$
动态规划	$O(n)$

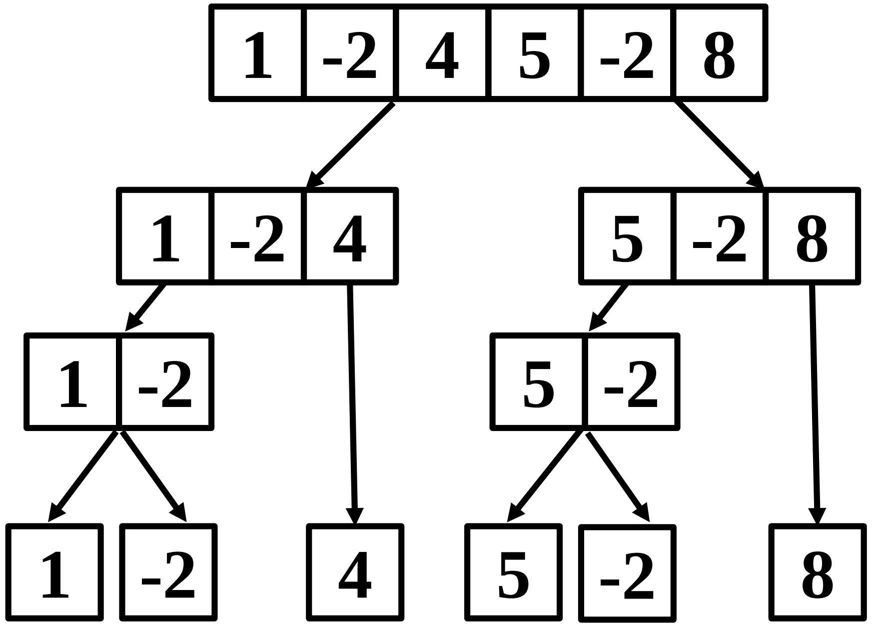


子问题相互独立

算法比较

基于枚举：蛮力枚举、优化枚举

算法名称	时间复杂度
蛮力枚举	$O(n^3)$
优化枚举	$O(n^2)$
分而治之	$O(n\log n)$
动态规划	$O(n)$



子问题相互独立

算法比较

算法名称	时间复杂度
蛮力枚举	$O(n^3)$
优化枚举	$O(n^2)$
分而治之	$O(n\log n)$
动态规划	$O(n)$

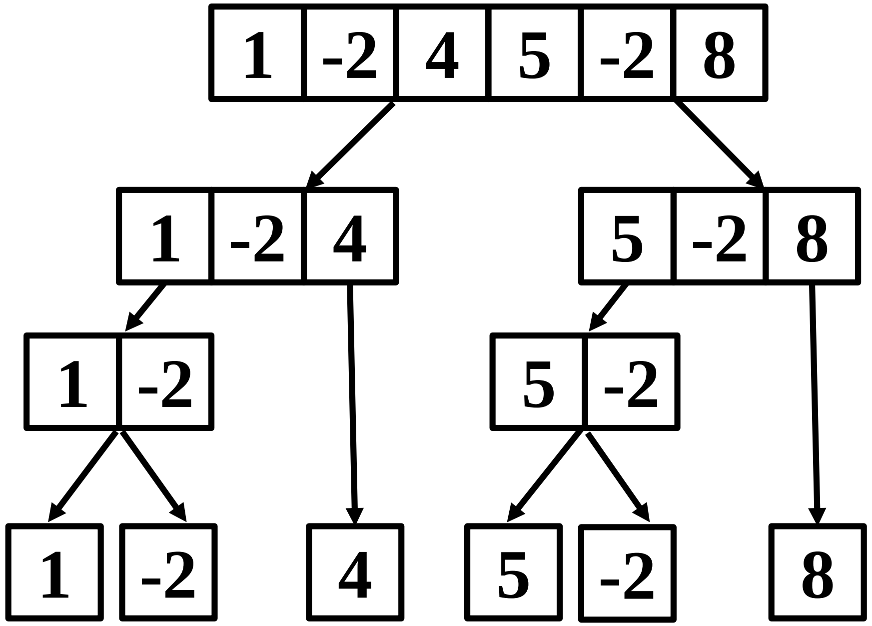
基于枚举：蛮力枚举、优化枚举

- 枚举区间

X	-1	-3	3	5	-4	3	2	-2	3

X	-1	-3	3	5	-4	3	2	-2	3

存在重叠子问题



子问题相互独立

枚举过程分析

- 两层枚举
  - 第1层：枚举位置i作为区间开头

枚举过程分析

- 两层枚举
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾

$i = 1$ $j = 1$

X	4	2	3	-5	3	1

当前最大值$= 4, S(i,j) = 4$

枚举过程分析

- **两层枚举**
  - **第1层：枚举位置i作为区间开头**
  - **第2层：枚举位置j作为区间结尾**

$i = 1$ $j = 2$

X						
	4	2	3	-5	3	1

当前最大值$= 6$, $S(i, j) = 6$

枚举过程分析

- 两层枚举
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾

$i = 1$ $\quad$ $j = 3$

X						
	4	2	3	-5	3	1

当前最大值$= 9$, $S(i, j) = 9$

枚举过程分析

- 两层枚举
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾

当前最大值$= 9,\ S(i, j) = 4$

枚举过程分析

- 两层枚举
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾

$i = 1$ $\qquad$ $j = 5$

X	4	2	3	-5	3	1

当前最大值$= 9$, $S(i, j) = 7$

枚举过程分析

- 两层枚举
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾

$i = 1$ $\qquad\qquad$ $j = 6$

X	4	2	3	-5	3	1

当前最大值$= 9$, $S(i, j) = 8$

枚举过程分析

- 两层枚举
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾

$i = 1$　　　　　　　　$j = 6$

X	4	2	3	-5	3	1

当前最大值$= 9$

D_i：以$X[i]$开头的最大子数组

枚举过程分析

- **两层枚举**
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾

X						
	4	2	3	-5	3	1

$$\underbrace{4\quad 2\quad 3}_{D_1 = 9}$$

$i = 1$　　　　$j = \cdots$

X						
	4	2	3	-5	3	1

当前最大值$= 9$

枚举过程分析

- 两层枚举
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾

X: $\underbrace{4\quad 2\quad 3}_{D_1 = 9}\quad -5\quad 3\quad 1$

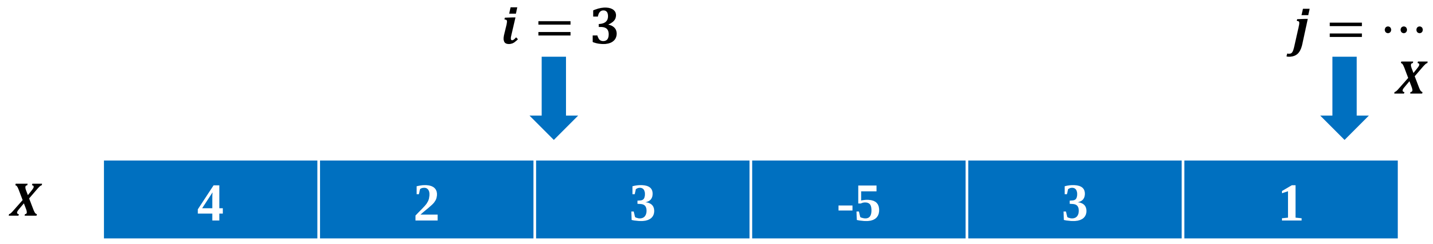
X: $4\quad \underbrace{2\quad 3}_{D_2 = 5}\quad -5\quad 3\quad 1$

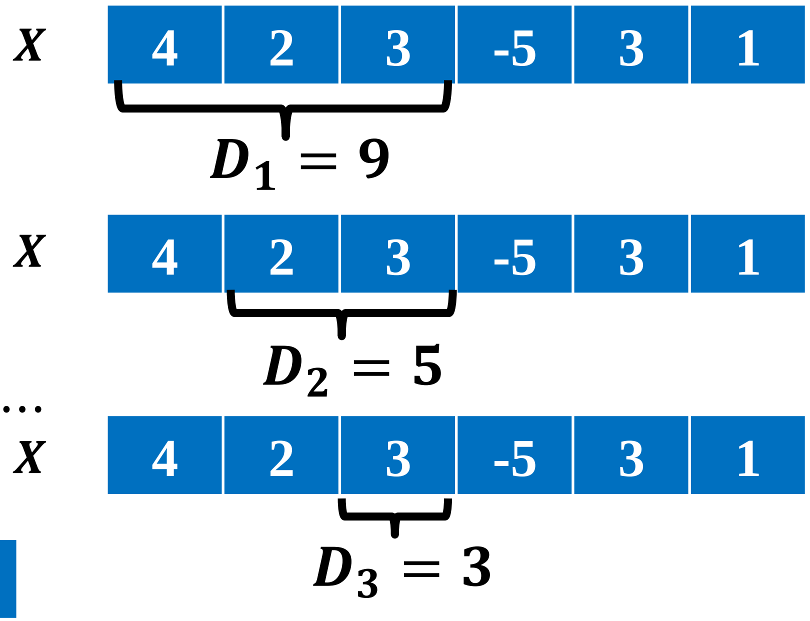
$i = 2$ ↓ (at the second element), $j = \cdots$ ↓ (at the last element)

X	4	2	3	-5	3	1

枚举过程分析

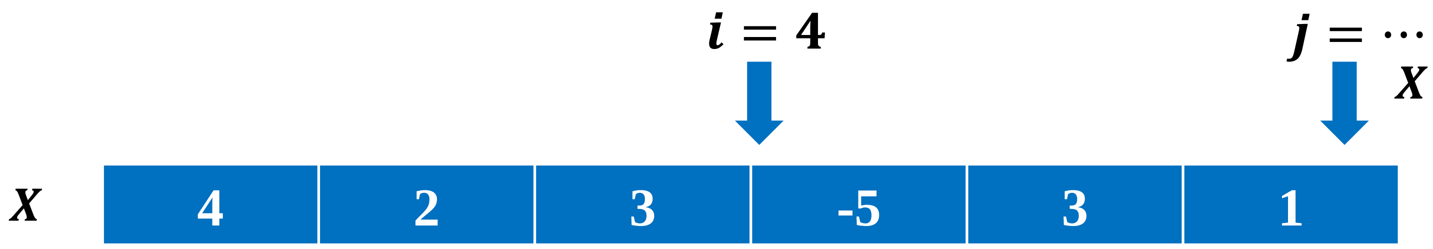
- 两层枚举
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾

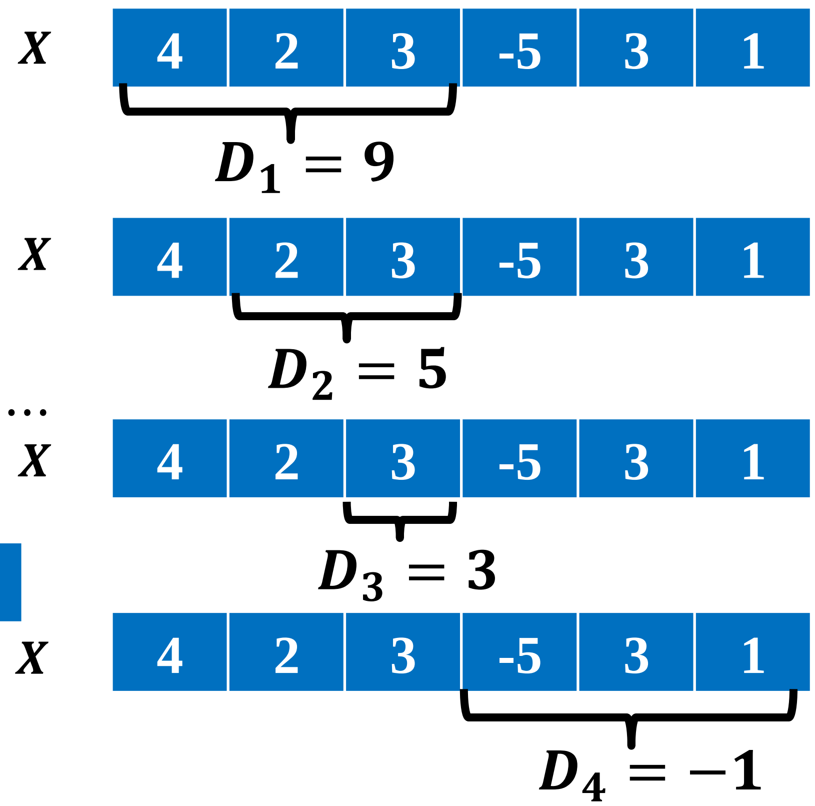




枚举过程分析

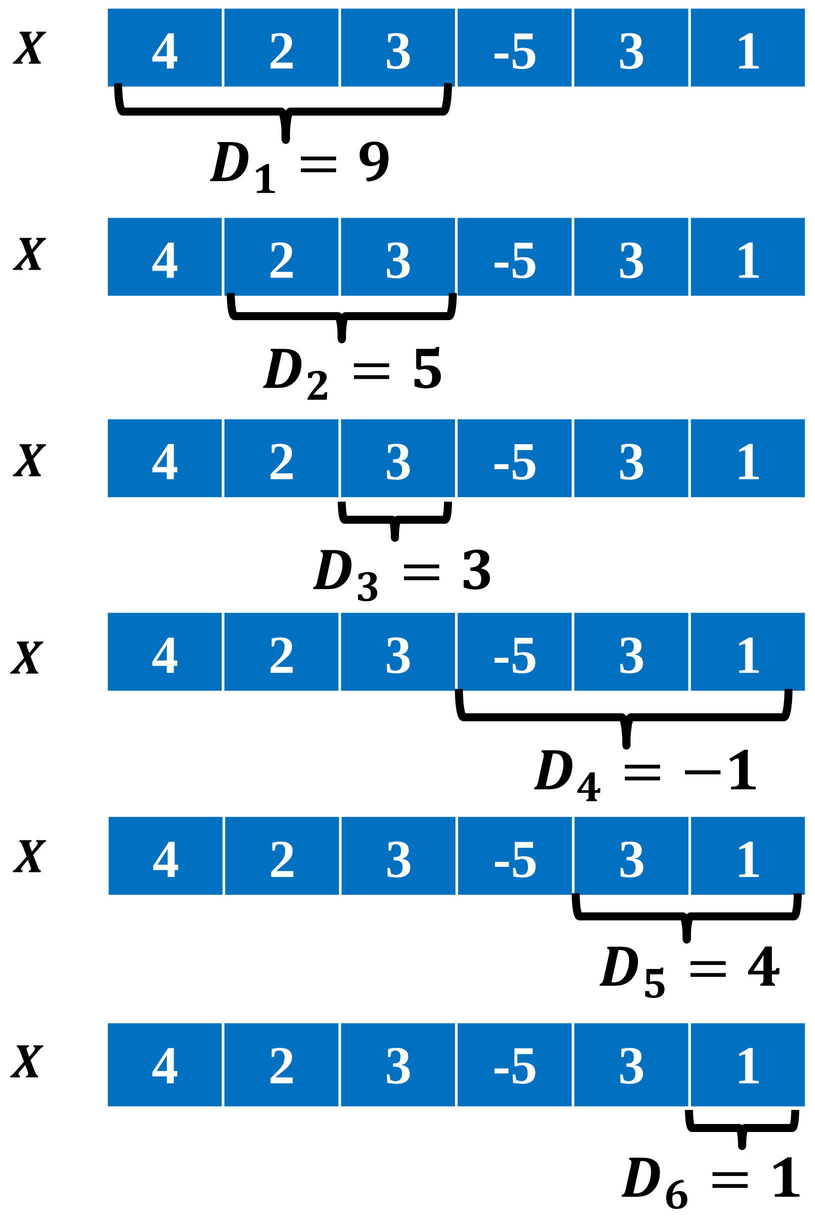
- 两层枚举
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾





枚举过程分析

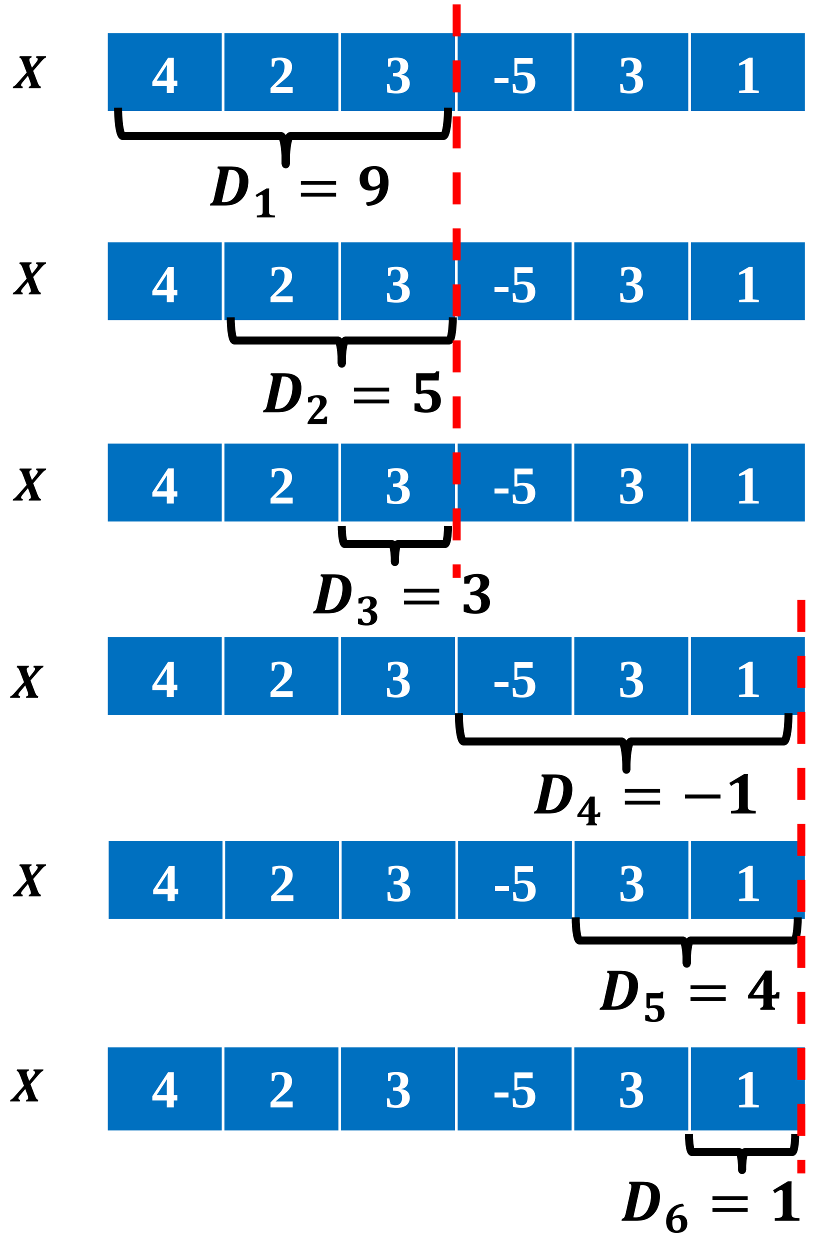
- 两层枚举
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾



枚举过程分析

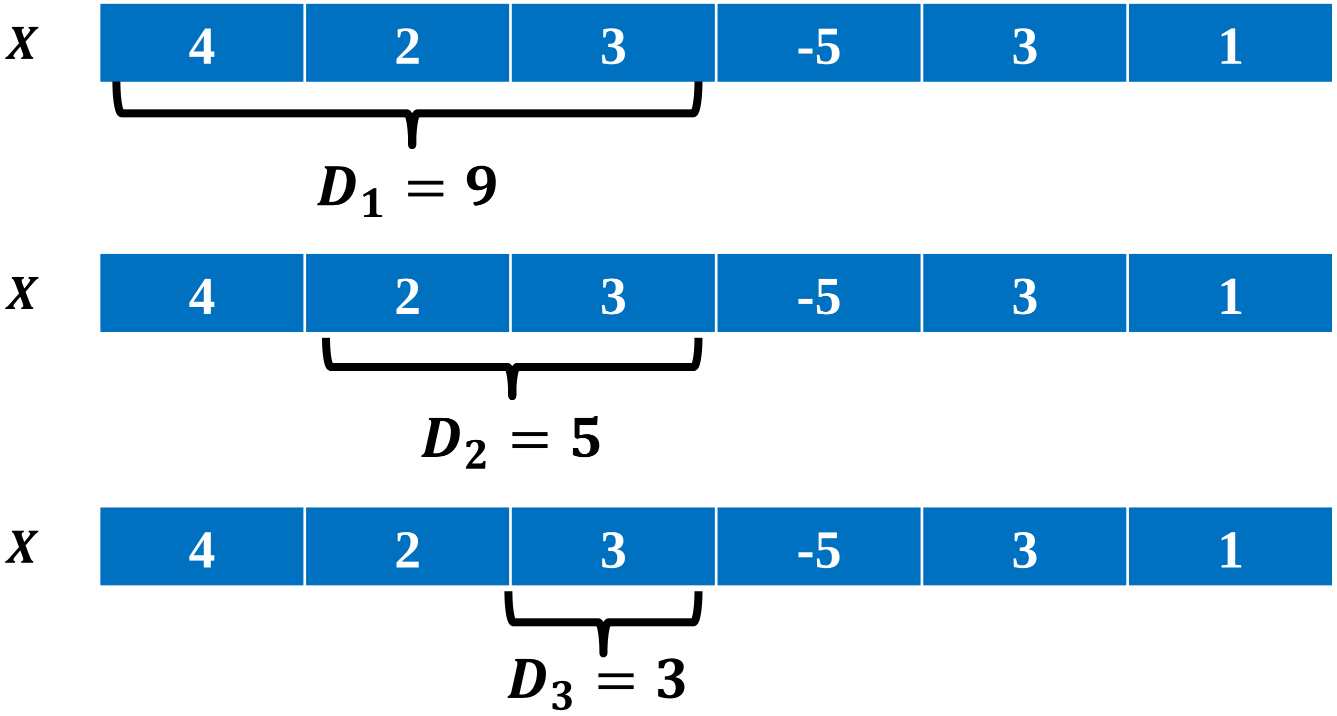
- 两层枚举
  - 第1层：枚举位置i作为区间开头
  - 第2层：枚举位置j作为区间结尾

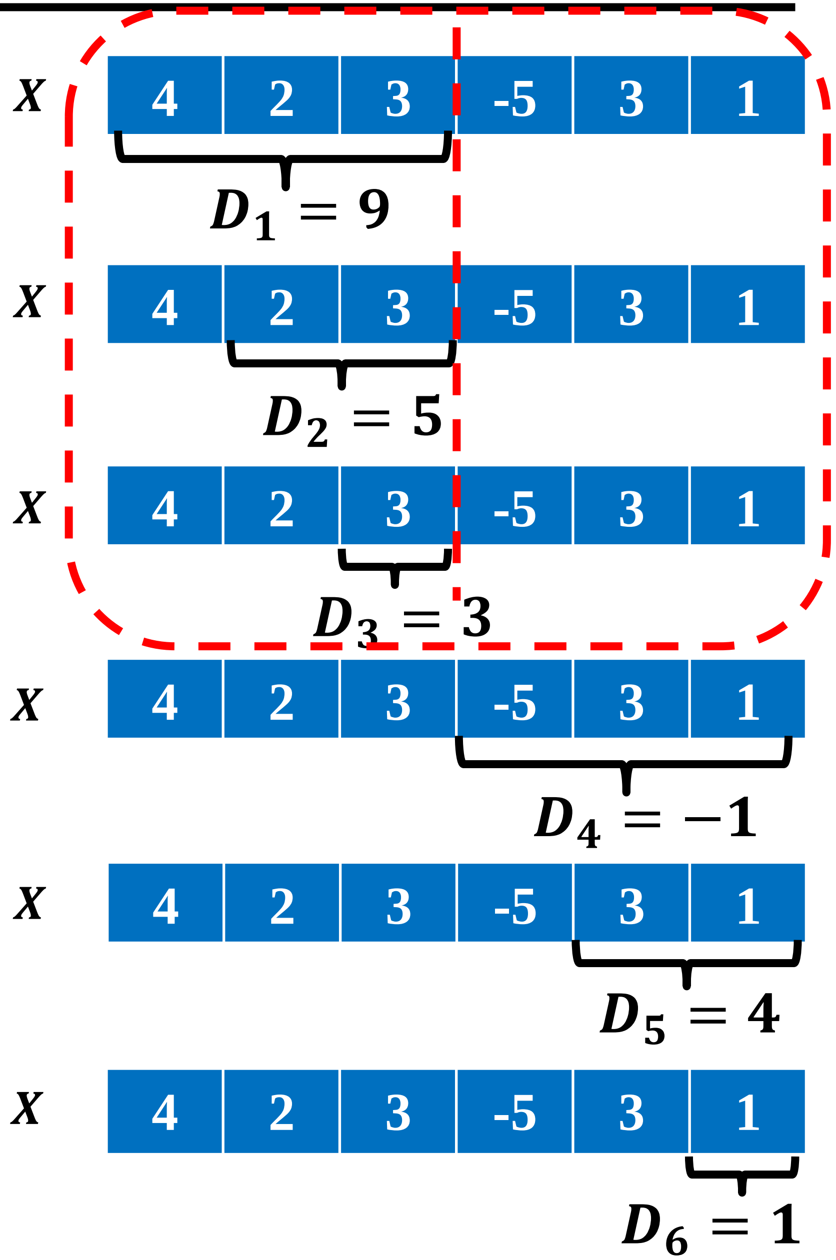
- 观察D_i
  - 结尾相同：D_1, D_2, D_3
  - 结尾不同：D_3, D_4



规律观察

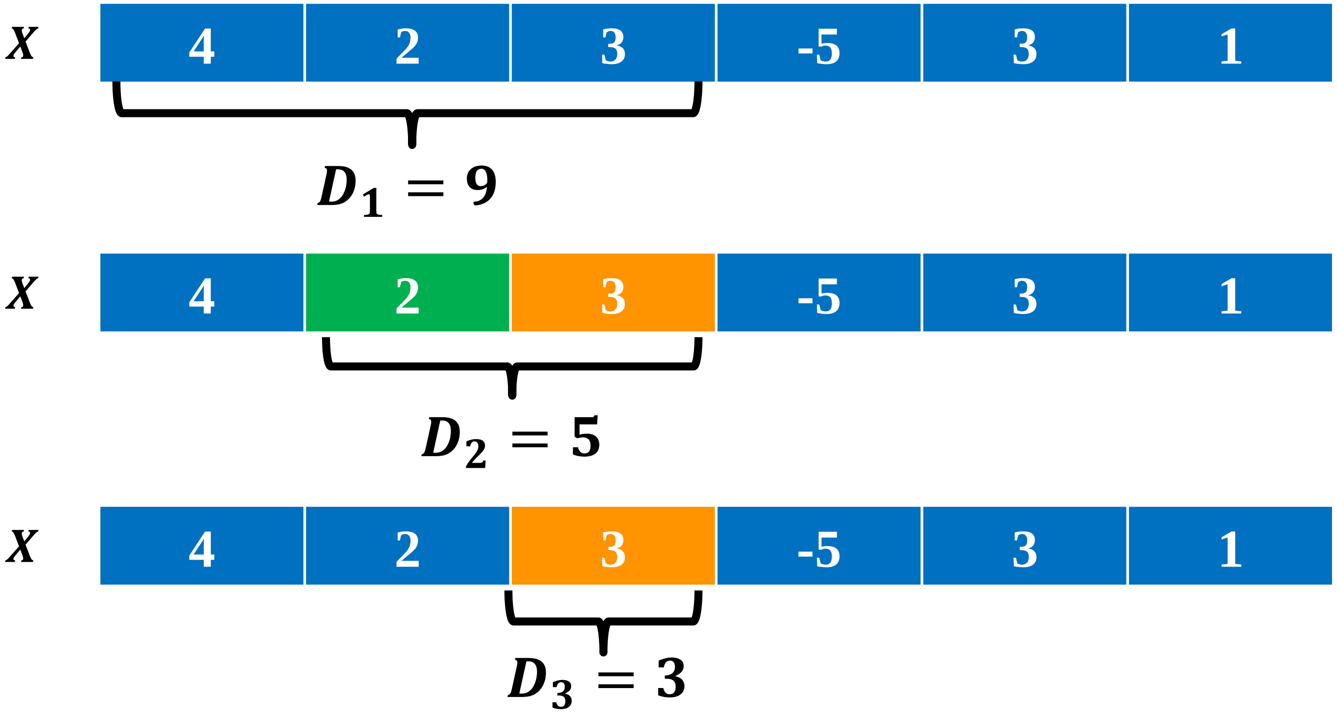
- 结尾位置相同
  - D_1, D_2, D_3

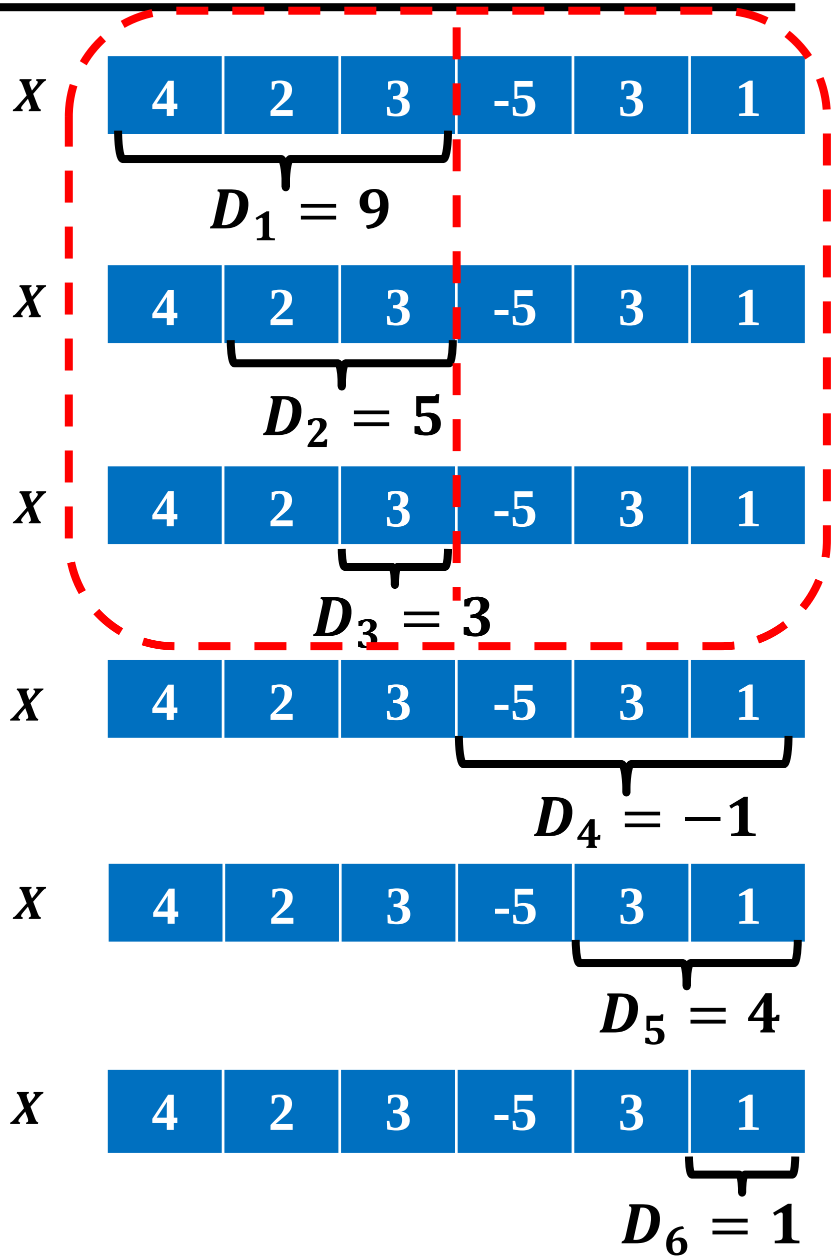




规律观察

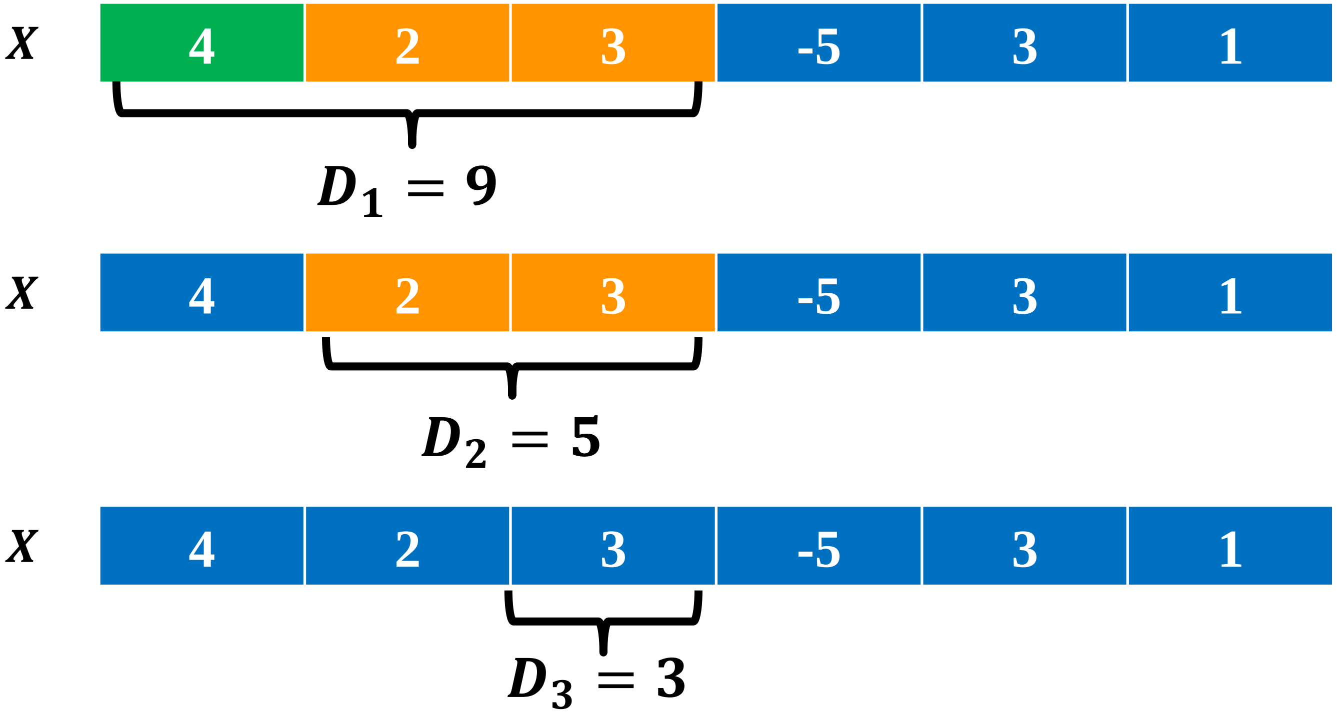
- 结尾位置相同
  - D_1, D_2, D_3
    - $D_2 = X[2] + D_3$

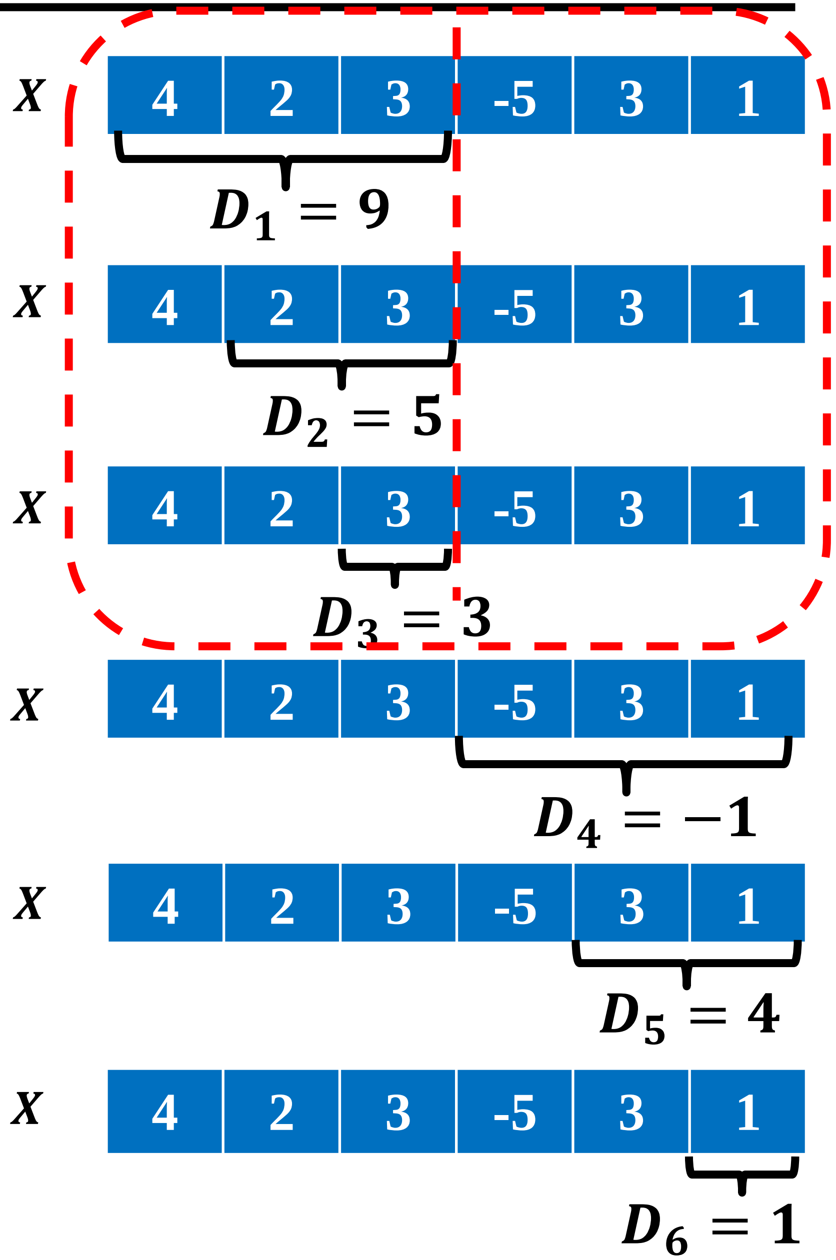




规律观察

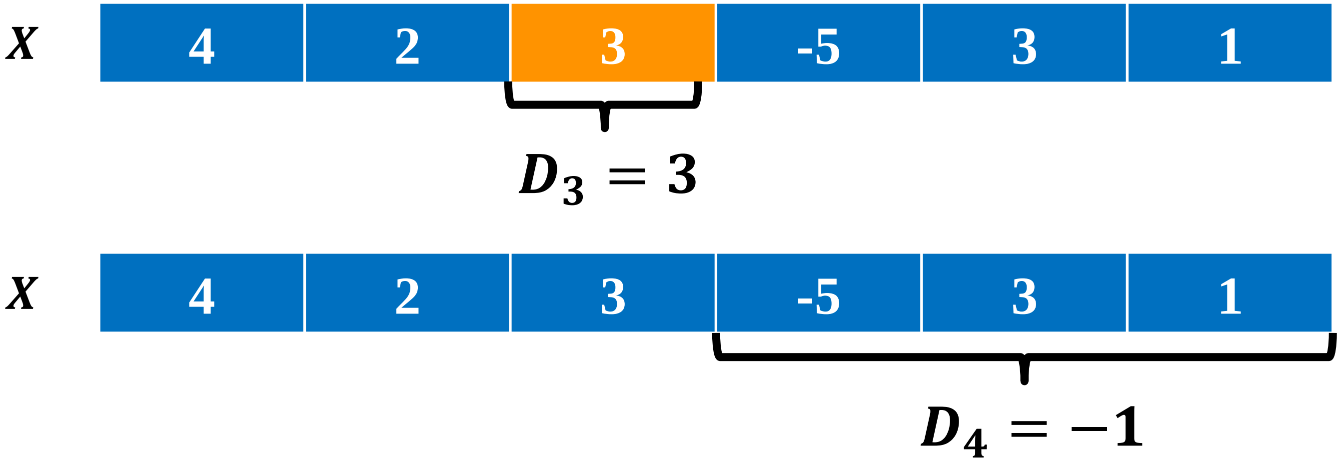
- 结尾位置相同
  - D_1, D_2, D_3
    - $D_2 = X[2] + D_3$
    - $D_1 = X[1] + D_2$

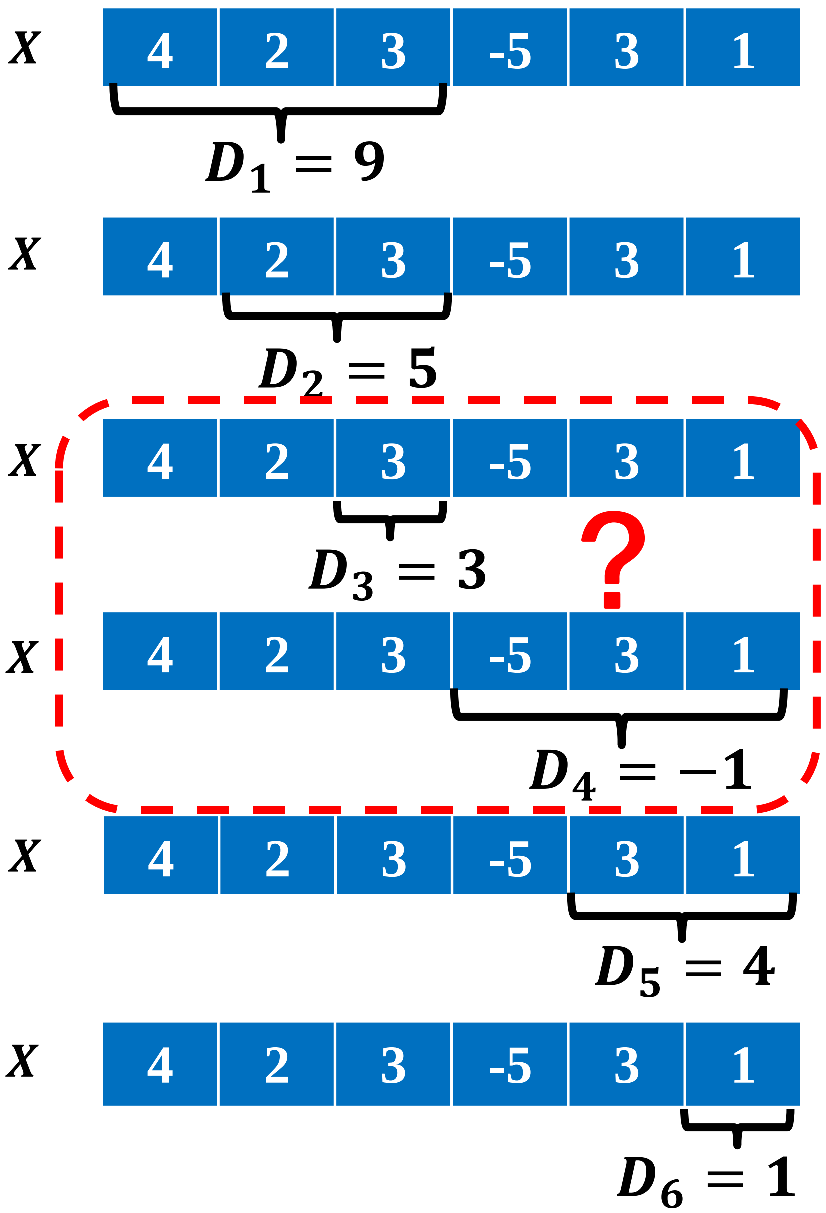




规律观察

- 结尾位置不同
  - D_3, D_4
    - $D_3 = X[3]$





规律观察

- 结尾位置不同
  - D_3, D_4
    - $D_3 = X[3]$
    - $D_4 < 0$

$$X \quad \begin{array}{|c|c|c|c|c|c|}\hline 4 & 2 & \underbrace{3}_{D_3 = 3} & -5 & 3 & 1 \\ \hline\end{array}$$

$$X \quad \begin{array}{|c|c|c|c|c|c|}\hline 4 & 2 & 3 & \underbrace{-5 \;\; 3 \;\; 1}_{D_4 = -1} \\ \hline\end{array}$$

$$X \quad \underbrace{4 \;\; 2 \;\; 3}_{D_1 = 9} \;\; -5 \;\; 3 \;\; 1$$

$$X \quad 4 \;\; \underbrace{2 \;\; 3}_{D_2 = 5} \;\; -5 \;\; 3 \;\; 1$$

$$X \quad 4 \;\; 2 \;\; \underbrace{3}_{D_3 = 3} \;\; -5 \;\; 3 \;\; 1$$

?

$$X \quad 4 \;\; 2 \;\; 3 \;\; \underbrace{-5 \;\; 3 \;\; 1}_{D_4 = -1}$$

$$X \quad 4 \;\; 2 \;\; 3 \;\; -5 \;\; \underbrace{3 \;\; 1}_{D_5 = 4}$$

$$X \quad 4 \;\; 2 \;\; 3 \;\; -5 \;\; 3 \;\; \underbrace{1}_{D_6 = 1}$$

规律描述

- D_i：以$X[i]$开头的最大子数组和

- 情况1：$D_{i+1} > 0$时

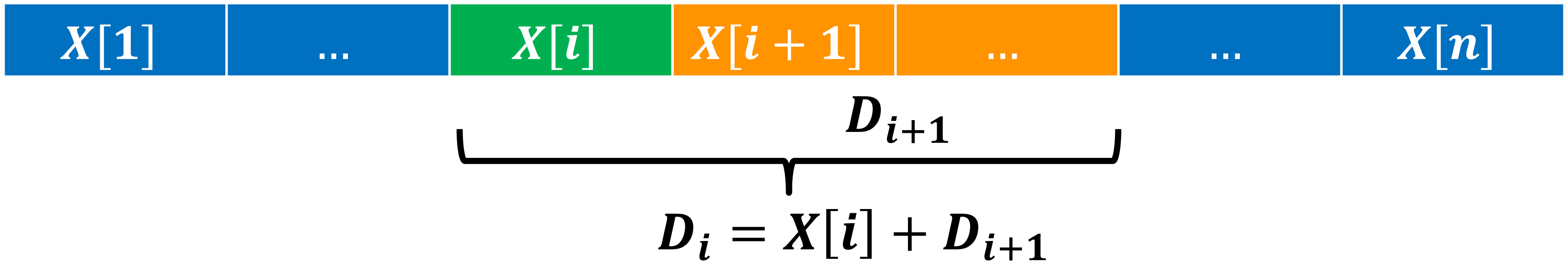
$X[1]$	…	$X[i]$	$X[i+1]$	…	…	$X[n]$

$$D_i = X[i] + \overbrace{X[i+1] + \dots}^{D_{i+1}} = X[i] + D_{i+1}$$

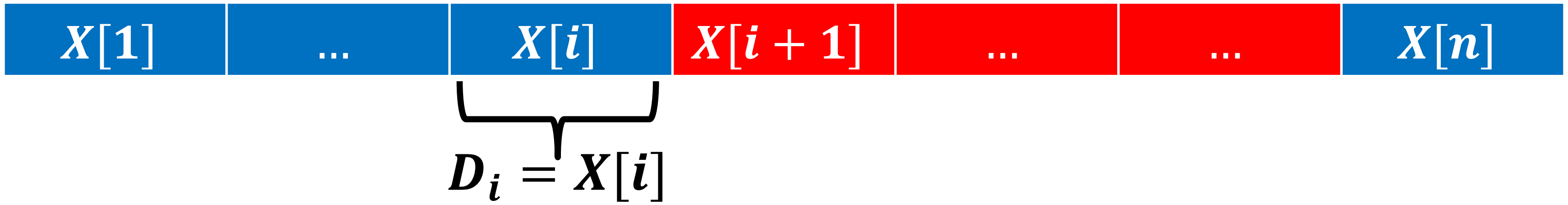
规律描述

- D_i：以$X[i]$开头的最大子数组和

- 情况1： $D_{i+1} > 0$时

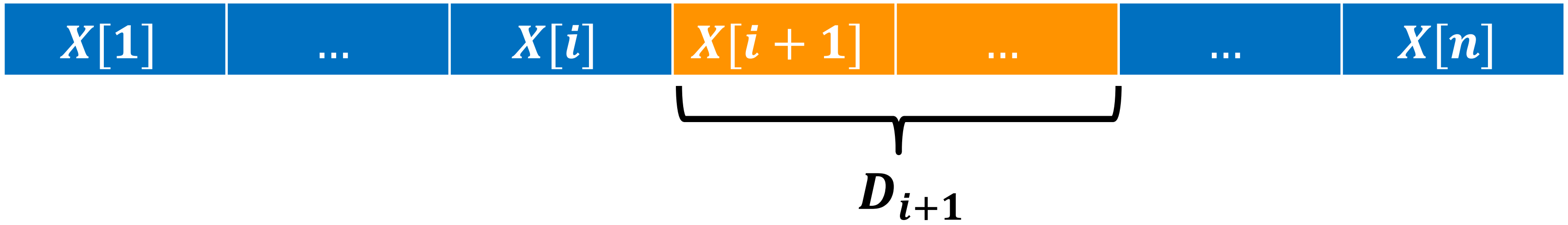


- 情况2：$D_{i+1} \leq 0$时



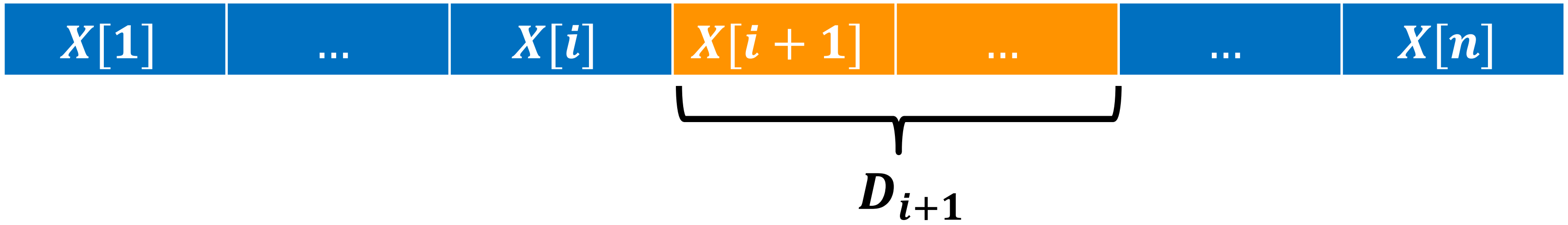
规律证明：情况1

- D_{i+1}：以$X[i+1]$开头的最大子数组和（$D_{i+1} > 0$）

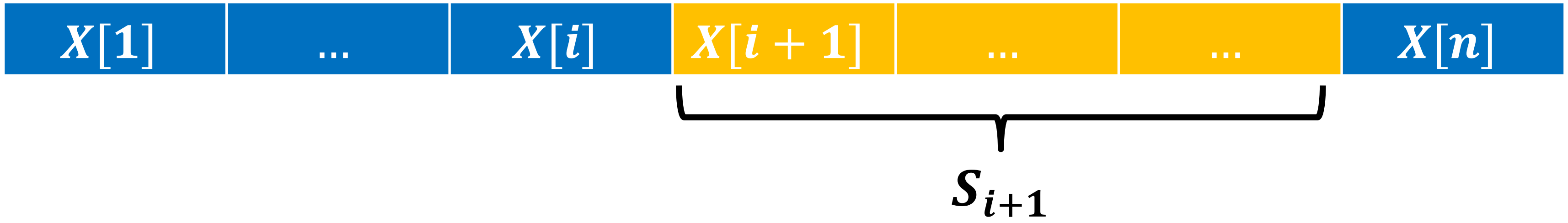


规律证明：情况1

- D_{i+1}：以$X[i+1]$开头的最大子数组和（$D_{i+1} > 0$）

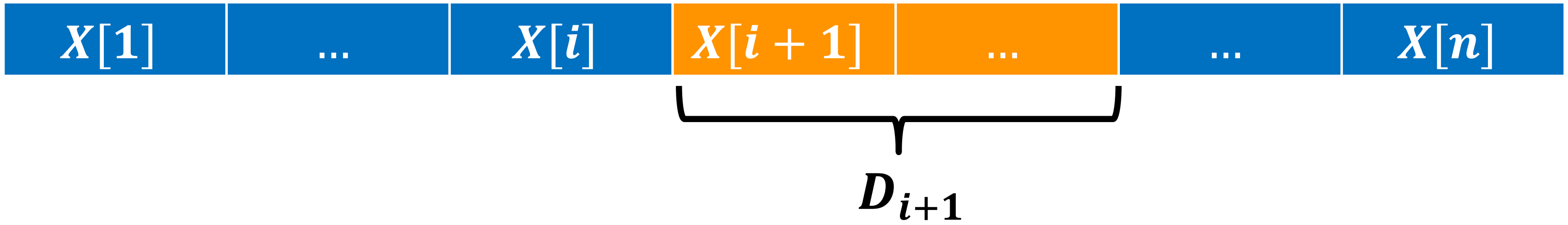


- S_{i+1}：以$X[i+1]$开头的任一子数组和

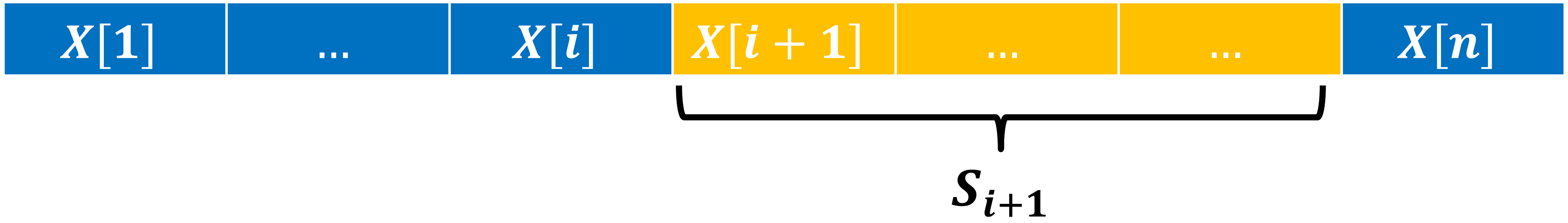


规律证明：情况1

- D_{i+1}：以$X[i+1]$开头的**最大**子数组和（$D_{i+1}>0$）

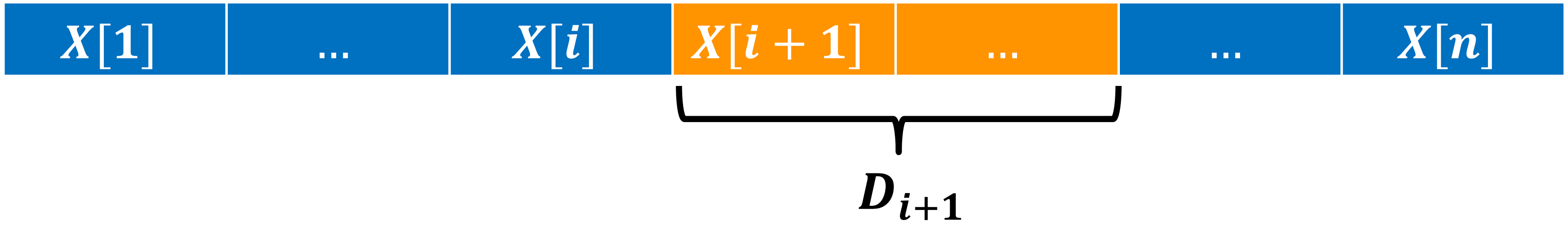


- S_{i+1}：以$X[i+1]$开头的**任一**子数组和

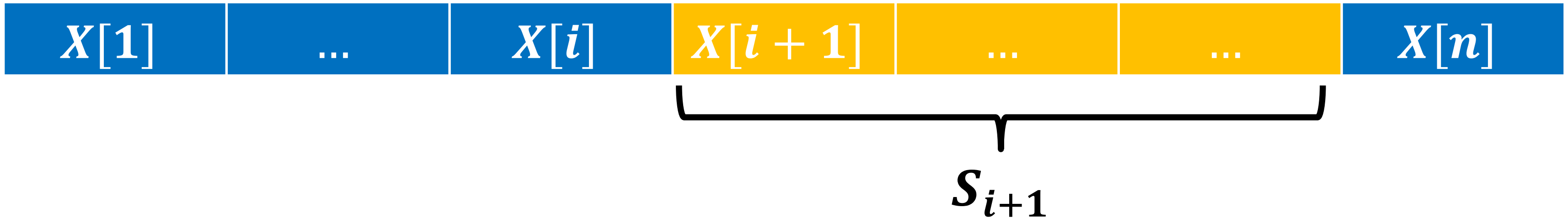


规律证明：情况1

- D_{i+1}：以$X[i+1]$开头的**最大**子数组和（$D_{i+1}>0$）

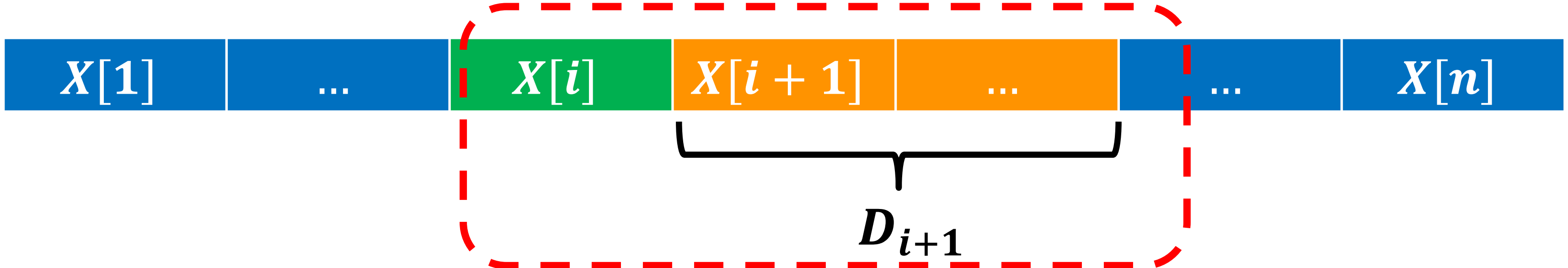


- S_{i+1}：以$X[i+1]$开头的**任一**子数组和（$S_{i+1} \le D_{i+1}$）

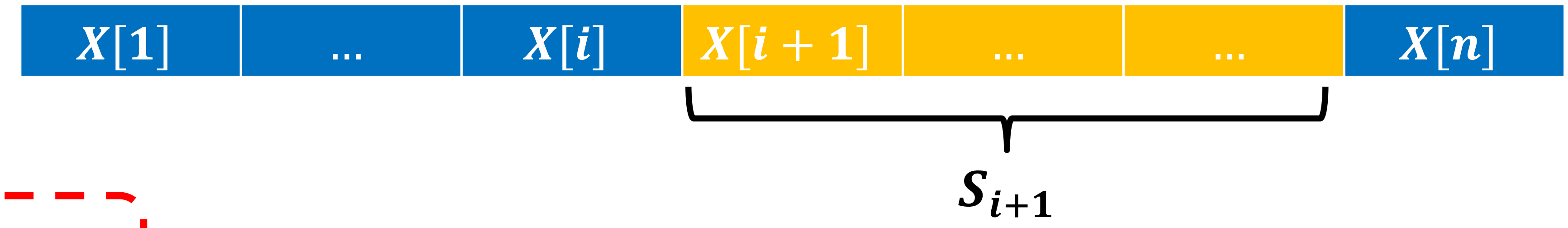


规律证明：情况1

- D_{i+1}：以$X[i+1]$开头的最大子数组和（$D_{i+1}>0$）



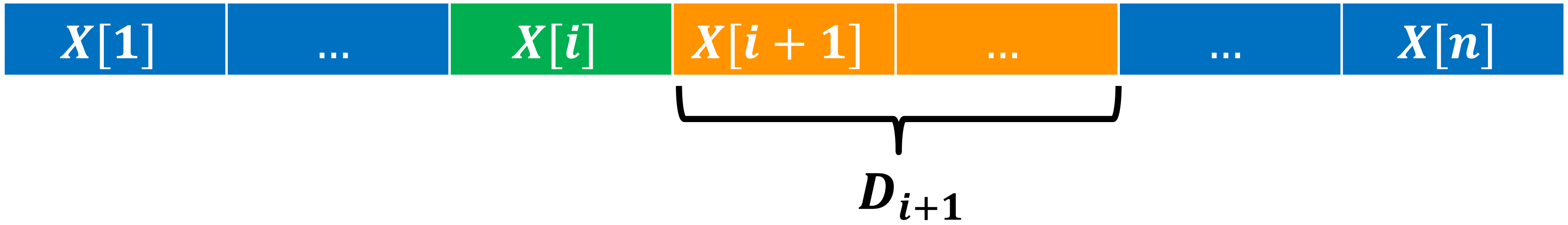
- S_{i+1}：以$X[i+1]$开头的任一子数组和（$S_{i+1} \le D_{i+1}$）



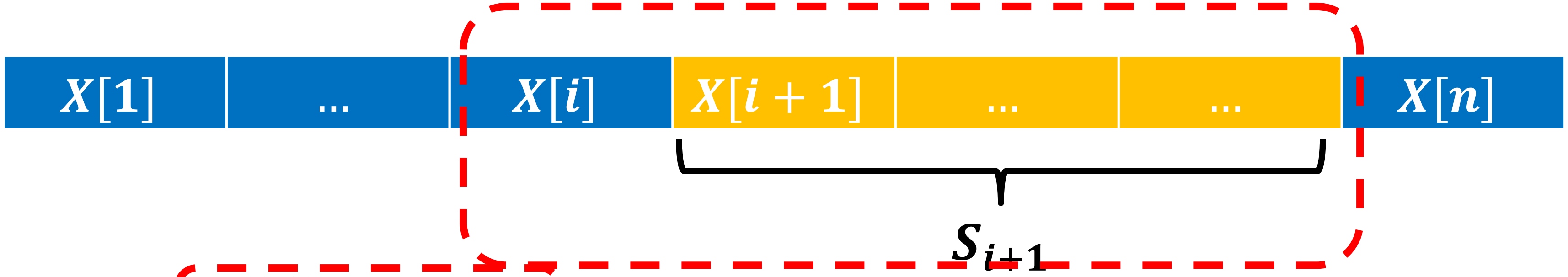
- $\boxed{X[i]+D_{i+1}}$

规律证明：情况1

- D_{i+1}：以$X[i+1]$开头的最大子数组和（$D_{i+1} > 0$）



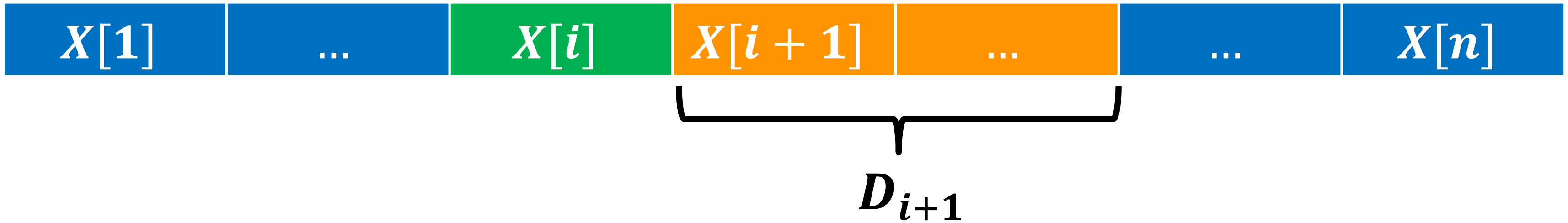
- S_{i+1}：以$X[i+1]$开头的任一子数组和（$S_{i+1} \le D_{i+1}$）



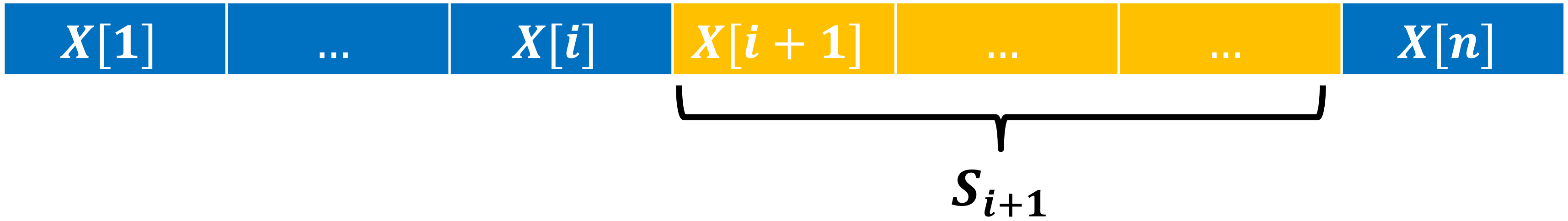
- $X[i]+D_{i+1}$ $\boxed{X[i]+S_{i+1}}$

规律证明：情况1

- D_{i+1}：以$X[i+1]$开头的最大子数组和（$D_{i+1} > 0$）



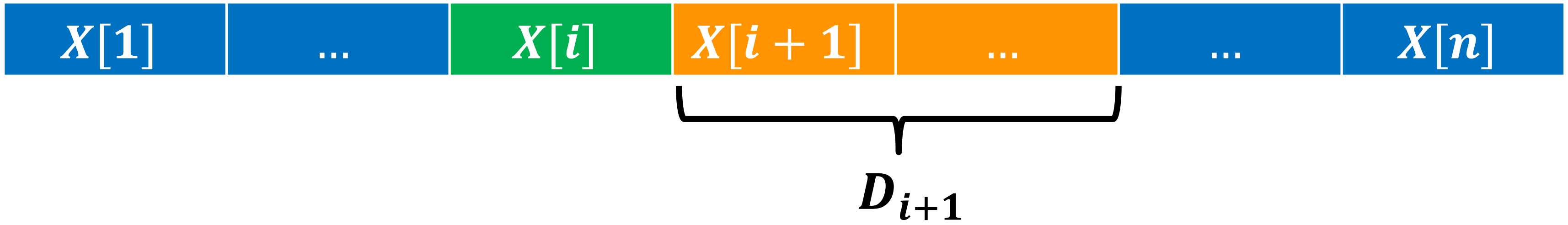
- S_{i+1}：以$X[i+1]$开头的任一子数组和（$S_{i+1} \leq D_{i+1}$）



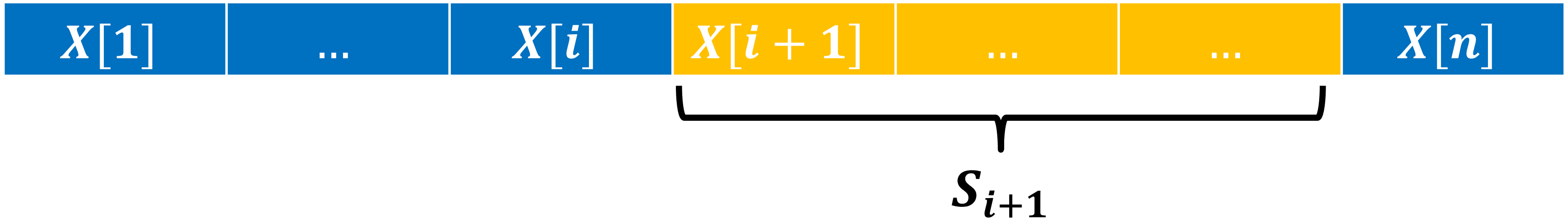
- $X[i] + D_{i+1} \geq X[i] + S_{i+1}$

规律证明：情况1

- D_{i+1}：以$X[i+1]$开头的最大子数组和（$D_{i+1} > 0$）



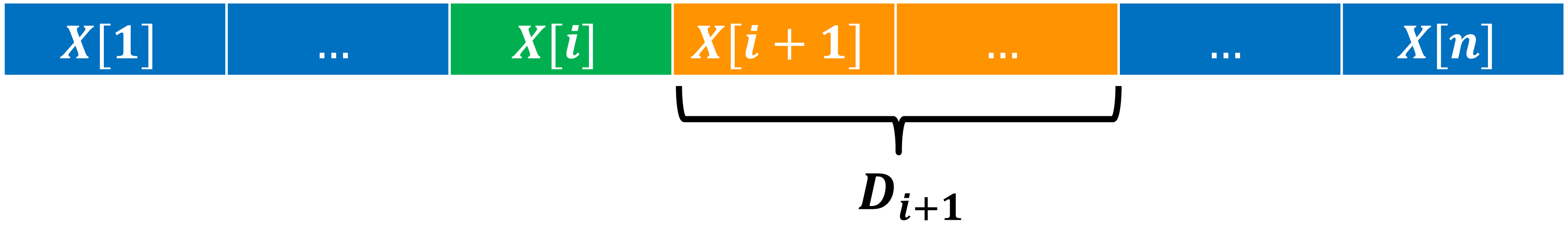
- S_{i+1}：以$X[i+1]$开头的任一子数组和（$S_{i+1} \le D_{i+1}$）



- $X[i] + D_{i+1} \ge X[i] + S_{i+1}$
- $D_i = X[i] + D_{i+1}$

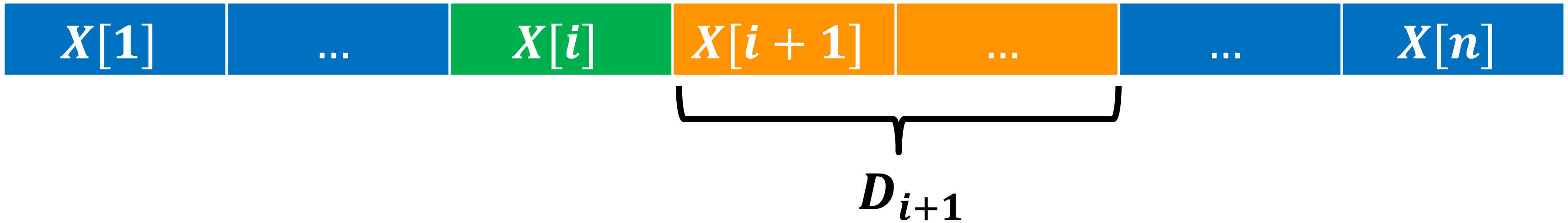
规律证明：情况2

- D_{i+1}：以$X[i+1]$开头的最大子数组和（$D_{i+1} \le 0$）

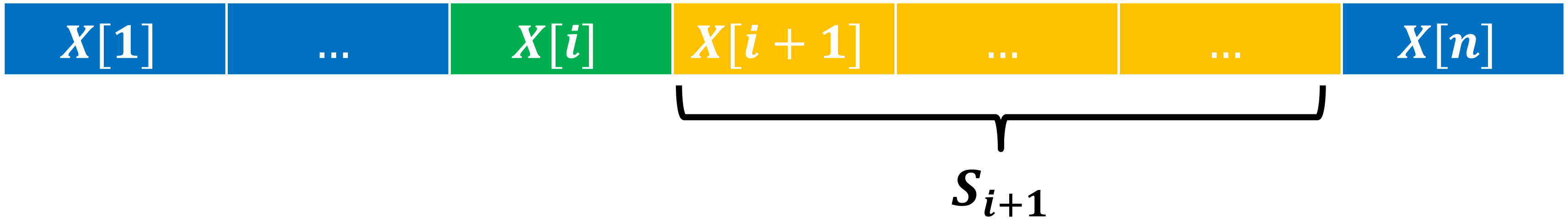


规律证明：情况2

- D_{i+1}：以$X[i+1]$开头的最大子数组和（$D_{i+1} \leq 0$）

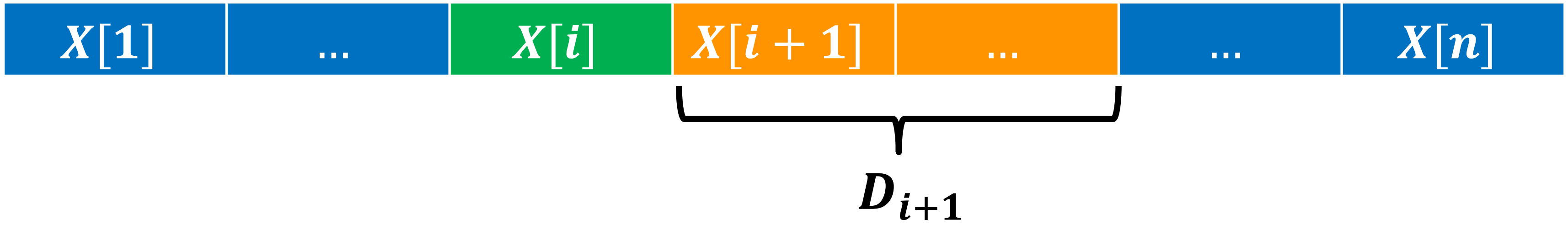


- S_{i+1}：以$X[i+1]$开头的任一子数组和（$S_{i+1} \leq D_{i+1}$）

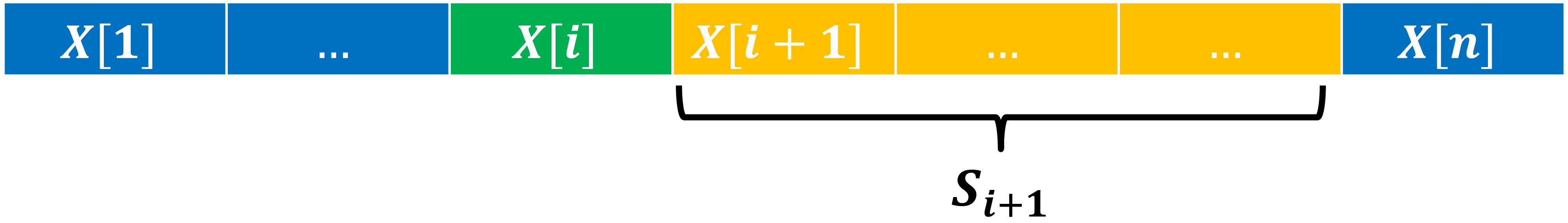


规律证明：情况2

- D_{i+1}：以$X[i+1]$开头的最大子数组和（$D_{i+1} \leq 0$）



- S_{i+1}：以$X[i+1]$开头的任一子数组和（$S_{i+1} \leq D_{i+1}$）



- $X[i] + S_{i+1} \leq X[i] + D_{i+1}$

规律证明：情况2

- D_{i+1}：以$X[i+1]$开头的最大子数组和（$D_{i+1} \le 0$）

$$X[1] \quad \ldots \quad X[i] \quad \underbrace{X[i+1] \quad \ldots}_{D_{i+1}} \quad \ldots \quad X[n]$$

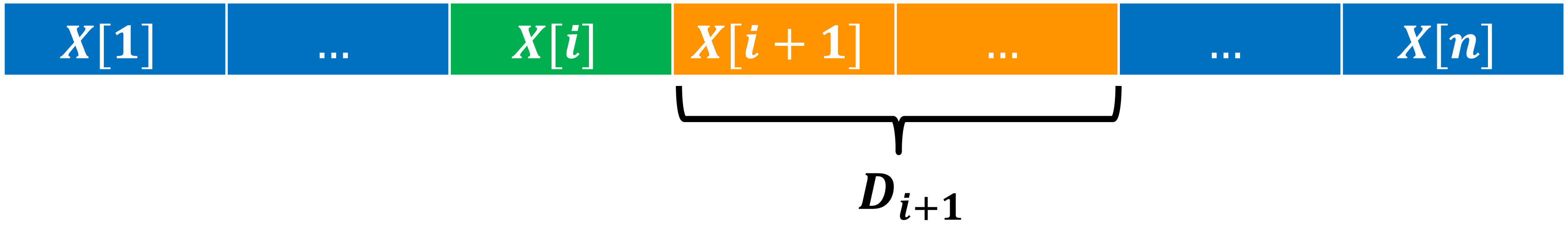
- S_{i+1}：以$X[i+1]$开头的任一子数组和（$S_{i+1} \le D_{i+1}$）

$$X[1] \quad \ldots \quad X[i] \quad \underbrace{X[i+1] \quad \ldots \quad \ldots}_{S_{i+1}} \quad X[n]$$

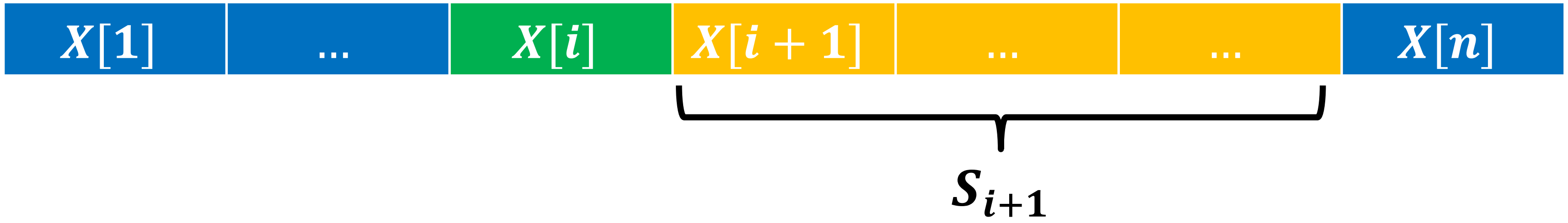
- $X[i] + S_{i+1} \le X[i] + D_{i+1} \le X[i]$

规律证明：情况2

- D_{i+1}：以$X[i+1]$开头的最大子数组和（$D_{i+1} \leq 0$）



- S_{i+1}：以$X[i+1]$开头的任一子数组和（$S_{i+1} \leq D_{i+1}$）



- $X[i] + S_{i+1} \leq X[i] + D_{i+1} \leq X[i]$
- $D_i = X[i]$

问题结构分析

- 给出问题表示
  - $D[i]$: 以$X[i]$开头的最大子数组和
- 明确原始问题
  - $S_{max} = \max_{1 \le i \le n}\{D[i]\}$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- $D[i]$：以$X[i]$开头的**最大**子数组和
- 情况1：$D[i+1] > 0$

$X[1]$	…	$X[i]$	$X[i+1]$	…	…	$X[n]$

$D[i+1]$

$D[i] = X[i] + D[i+1]$

- 情况2：$D[i+1] \le 0$

$X[1]$	…	$X[i]$	$X[i+1]$	…	…	$X[n]$

$D[i] = X[i]$

- 问题结构分析
- 递推关系建立
- 自底向上计算
- 最优方案追踪

递推关系建立：分析最优（子）结构

- $D[i]$：以$X[i]$开头的**最大**子数组和
- 情况1：$D[i+1] > 0$

$X[1]$	…	$X[i]$	$X[i+1]$	…	…	$X[n]$

$D[i+1]$

$$\boxed{D[i]} = X[i] + \boxed{D[i+1]}$$

最优子结构

- 情况2：$D[i+1] \leq 0$

$X[1]$	…	$X[i]$	$X[i+1]$	…	…	$X[n]$

$$D[i] = X[i]$$

- 问题结构分析
- 递推关系建立
- 自底向上计算
- 最优方案追踪

递推关系建立：构造递推公式

$X[1]$	…	$X[i]$	$X[i+1]$	…	…	$X[n]$

$D[i+1]$

$D[i] = X[i] + D[i+1]$

$X[1]$	…	$X[i]$	$X[i+1]$	…	…	$X[n]$

$D[i] = X[i]$

- $$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \leq 0 \end{cases}$$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

已知

	1	2	…	i	i+1	…	$n-1$	n
X	$X[1]$	$X[2]$		$X[i]$				$X[n]$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

- 初始化
  - $D[n] = X[n]$

已知

	1	2	…	i	i+1	…	$n-1$	n
X	$X[1]$	$X[2]$		$X[i]$				$X[n]$

	1	2	…	i	i+1	…	$n-1$	n
D								$X[n]$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

- 初始化
  - $D[n] = X[n]$
- 递推公式
  - $D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \le 0 \end{cases}$

已知

	1	2	…	i	i+1	…	$n-1$	n
X	$X[1]$	$X[2]$		$\boxed{X[i]}$				$X[n]$

	1	2	…	i	i+1	…	$n-1$	n
D								$X[n]$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

- 初始化
  - $D[n] = X[n]$
- 递推公式
  - $D[i] = \begin{cases} X[i] + \color{red}{D[i+1]}, & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \le 0 \end{cases}$

已知

	1	2	…	i	i+1	…	$n-1$	n
X	$X[1]$	$X[2]$		$X[i]$				$X[n]$

	1	2	…	i	i+1	…	$n-1$	n
D					$\boxed{D[i+1]}$			$X[n]$

问题结构分析
↓
递推关系建立
↓
自底向上计算
↓
最优方案追踪

自底向上计算：确定计算顺序

- 初始化
  - $D[n] = X[n]$
- 递推公式
  - $D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \le 0 \end{cases}$

已知

	1	2	…	i	i+1	…	$n-1$	n
X	$X[1]$	$X[2]$		$X[i]$				$X[n]$

	1	2	…	i	i+1	…	$n-1$	n
D				$\boxed{D[i]}$	$D[i+1]$			$X[n]$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：依次求解问题

- **初始化**
  - $D[n] = X[n]$
- **递推公式**
  - $$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \le 0 \end{cases}$$

已知

	1	2	…	i	i+1	…	$n-1$	n
X	$X[1]$	$X[2]$		$X[i]$				$X[n]$

	1	2	…	i	i+1	…	$n-1$	n
D								$X[n]$

自底向上计算

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：记录决策过程

- 构造追踪数组$Rec[1..n]$
  - 情况1：结尾相同
    - $Rec[i] = Rec[i+1]$

	1	2	…	i	…		$n-1$	n
X	$X[1]$	$X[2]$		$X[i]$				$X[n]$

$D[i+1]$

$D[i]$

	1	2	…	i	i+1	…	$n-1$	n
Rec								n

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：记录决策过程

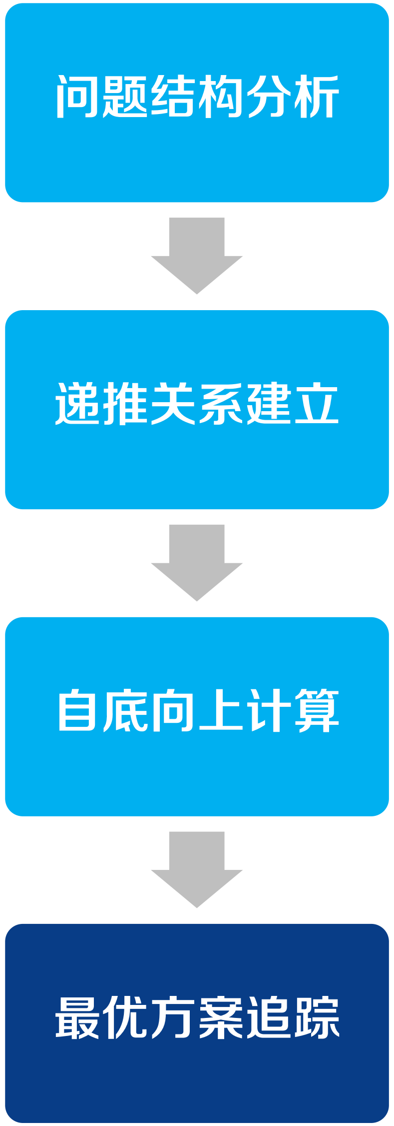
- 构造追踪数组$Rec[1..n]$
  - 情况1：结尾相同
    - $Rec[i] = Rec[i+1]$

	1	2	…	i	…		$n-1$	n
X	$X[1]$	$X[2]$		$X[i]$				$X[n]$

$D[i+1]$

$D[i]$

	1	2	…	i	i+1	…	$n-1$	n
Rec								n



最优方案追踪：记录决策过程

- 构造追踪数组$Rec[1..n]$
  - 情况2：结尾不同
    - $Rec[i] = i$

	1	2	…	i	…		$n-1$	n
X	$X[1]$	$X[2]$		$X[i]$				$X[n]$

$D[i]$

	1	2	…	i	i+1	…	$n-1$	n
Rec								n

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：输出最优方案

- 从子问题中查找最优解

	1	2	...	...	i		$n-1$	n
X	$X[1]$	$X[2]$			$X[i]$			$X[n]$

已求得

	1	2	...		i		$n-1$	n
D	$D[1]$	$D[2]$			$D[i]$			$D[n]$

	1	2	...		i	...	$n-1$	n
Rec								n

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：输出最优方案

- 从子问题中查找最优解

	1	2	…	…	i		$n-1$	n
X	$X[1]$	$X[2]$			$X[i]$			$X[n]$

	1	2	…		i		$n-1$	n
D	$D[1]$	$D[2]$			$D[i]$			$D[n]$

最优解

	1	2	…		i	…	$n-1$	n
Rec								n

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：输出最优方案

- 从子问题中查找最优解
- 最大子数组开头位置：i

	1	2	$\dots$	$\dots$	i		$n-1$	n
X	$X[1]$	$X[2]$			$X[i]$			$X[n]$

	1	2	$\dots$		i		$n-1$	n
D	$D[1]$	$D[2]$			$D[i]$			$D[n]$

	1	2	$\dots$		i	$\dots$	$n-1$	n
Rec								n

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

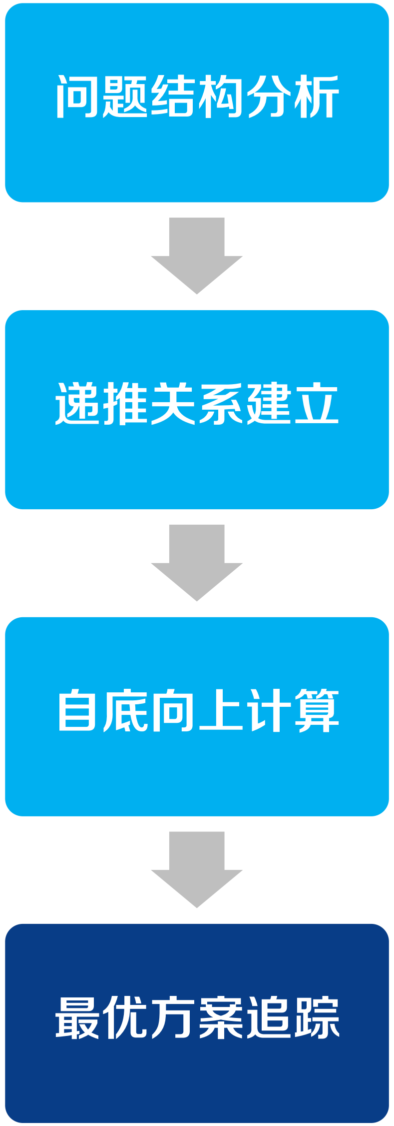
最优方案追踪：输出最优方案

- 从子问题中查找最优解
- 最大子数组开头位置：i
- 最大子数组结尾位置：$Rec[i]$

	1	2	…	…	i		$n-1$	n
X	$X[1]$	$X[2]$			$X[i]$			$X[n]$

	1	2	…		i		$n-1$	n
D	$D[1]$	$D[2]$			$D[i]$			$D[n]$

	1	2	…		i	…	$n-1$	n
Rec					$Rec[i]$			n



算法实例

i	1	2	3	4	5	6	7	8	9	10	11	12
X	1	-2	4	5	-2	8	3	-2	6	3	7	-1

i	1	2	3	4	5	6	7	8	9	10	11	12
D												

i	1	2	3	4	5	6	7	8	9	10	11	12
Rec												

算法实例

	i	1	2	3	4	5	6	7	8	9	10	11	12
X		1	-2	4	5	-2	8	3	-2	6	3	7	-1

	i	1	2	3	4	5	6	7	8	9	10	11	12
D													-1

初始化

	i	1	2	3	4	5	6	7	8	9	10	11	12
Rec													12

算法实例

i	1	2	3	4	5	6	7	8	9	10	11	12
X	1	-2	4	5	-2	8	3	-2	6	3	7	-1

i	1	2	3	4	5	6	7	8	9	10	11	12
D											7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \leq 0 \end{cases}$$

i	1	2	3	4	5	6	7	8	9	10	11	12
Rec												12

算法实例

X

i	1	2	3	4	5	6	7	8	9	10	11	12
	1	-2	4	5	-2	8	3	-2	6	3	7	-1

D

i	1	2	3	4	5	6	7	8	9	10	11	12
											7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \leq 0 \end{cases}$$

Rec

i	1	2	3	4	5	6	7	8	9	10	11	12
											11	12

算法实例

	i	1	2	3	4	5	6	7	8	9	10	11	12
X		1	-2	4	5	-2	8	3	-2	6	3	7	-1

	i	1	2	3	4	5	6	7	8	9	10	11	12
D											10	7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \le 0 \end{cases}$$

	i	1	2	3	4	5	6	7	8	9	10	11	12
Rec											11	11	12

算法实例

	i	1	2	3	4	5	6	7	8	9	10	11	12
X		1	-2	4	5	-2	8	3	-2	6	3	7	-1

	i	1	2	3	4	5	6	7	8	9	10	11	12
D										16	10	7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \leq 0 \end{cases}$$

	i	1	2	3	4	5	6	7	8	9	10	11	12
Rec										11	11	11	12

算法实例

	i	1	2	3	4	5	6	7	8	9	10	11	12
X		1	-2	4	5	-2	8	3	-2	6	3	7	-1

	i	1	2	3	4	5	6	7	8	9	10	11	12
D									14	16	10	7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \leq 0 \end{cases}$$

	i	1	2	3	4	5	6	7	8	9	10	11	12
Rec									11	11	11	11	12

算法实例

i	1	2	3	4	5	6	7	8	9	10	11	12
X	1	-2	4	5	-2	8	3	-2	6	3	7	-1

i	1	2	3	4	5	6	7	8	9	10	11	12
D							17	14	16	10	7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \leq 0 \end{cases}$$

i	1	2	3	4	5	6	7	8	9	10	11	12
Rec							11	11	11	11	11	12

算法实例

i	1	2	3	4	5	6	7	8	9	10	11	12
X	1	-2	4	5	-2	8	3	-2	6	3	7	-1

i	1	2	3	4	5	6	7	8	9	10	11	12
D						25	17	14	16	10	7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \leq 0 \end{cases}$$

i	1	2	3	4	5	6	7	8	9	10	11	12
Rec						11	11	11	11	11	11	12

算法实例

i	1	2	3	4	5	6	7	8	9	10	11	12
X	1	-2	4	5	-2	8	3	-2	6	3	7	-1

i	1	2	3	4	5	6	7	8	9	10	11	12
D					23	25	17	14	16	10	7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \leq 0 \end{cases}$$

i	1	2	3	4	5	6	7	8	9	10	11	12
Rec					11	11	11	11	11	11	11	12

算法实例

	i	1	2	3	4	5	6	7	8	9	10	11	12
X		1	-2	4	$\boxed{5}$	-2	8	3	-2	6	3	7	-1

	i	1	2	3	4	5	6	7	8	9	10	11	12
D					$\boxed{28}$	$\boxed{23}$	25	17	14	16	10	7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \le 0 \end{cases}$$

	i	1	2	3	4	5	6	7	8	9	10	11	12
Rec					$\boxed{11}$	11	11	11	11	11	11	11	12

算法实例

i	1	2	3	4	5	6	7	8	9	10	11	12
X	1	-2	4	5	-2	8	3	-2	6	3	7	-1

i	1	2	3	4	5	6	7	8	9	10	11	12
D			32	28	23	25	17	14	16	10	7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \leq 0 \end{cases}$$

i	1	2	3	4	5	6	7	8	9	10	11	12
Rec			11	11	11	11	11	11	11	11	11	12

算法实例

i	1	2	3	4	5	6	7	8	9	10	11	12
X	1	-2	4	5	-2	8	3	-2	6	3	7	-1

i	1	2	3	4	5	6	7	8	9	10	11	12
D		30	32	28	23	25	17	14	16	10	7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \leq 0 \end{cases}$$

i	1	2	3	4	5	6	7	8	9	10	11	12
Rec		11	11	11	11	11	11	11	11	11	11	12

算法实例

	i	1	2	3	4	5	6	7	8	9	10	11	12
X		1	-2	4	5	-2	8	3	-2	6	3	7	-1

	i	1	2	3	4	5	6	7	8	9	10	11	12
D		31	30	32	28	23	25	17	14	16	10	7	-1

$$D[i] = \begin{cases} X[i] + D[i+1], & if\ D[i+1] > 0 \\ X[i], & if\ D[i+1] \leq 0 \end{cases}$$

	i	1	2	3	4	5	6	7	8	9	10	11	12
Rec		11	11	11	11	11	11	11	11	11	11	11	12

算法实例

i	1	2	3	4	5	6	7	8	9	10	11	12
X	1	-2	4	5	-2	8	3	-2	6	3	7	-1

i	1	2	3	4	5	6	7	8	9	10	11	12
D	31	30	32	28	23	25	17	14	16	10	7	-1

最优解

i	1	2	3	4	5	6	7	8	9	10	11	12
Rec	11	11	11	11	11	11	11	11	11	11	11	12

算法实例

	i	1	2	3	4	5	6	7	8	9	10	11	12
X		1	-2	4	5	-2	8	3	-2	6	3	7	-1

	i	1	2	3	4	5	6	7	8	9	10	11	12
D		31	30	32	28	23	25	17	14	16	10	7	-1

最优解

	i	1	2	3	4	5	6	7	8	9	10	11	12
Rec		11	11	11	11	11	11	11	11	11	11	11	12

终止位置

算法实例

i	1	2	3	4	5	6	7	8	9	10	11	12
X	1	-2	4	5	-2	8	3	-2	6	3	7	-1

i	1	2	3	4	5	6	7	8	9	10	11	12
D	31	30	32	28	23	25	17	14	16	10	7	-1

最优解

i	1	2	3	4	5	6	7	8	9	10	11	12
Rec	11	11	11	11	11	11	11	11	11	11	11	12

终止位置

$$S = \{4, 5, -2, 8, 3, -2, 6, 3, 7\}$$

动态规划：伪代码

- **Max-Continuous-Subarray-DP(X, n)**

```
输入: 数组X, 数组长度n
输出: 最大子数组和S_max,子数组起止位置l, r
新建一维数组D[1..n]和Rec[1..n]
//初始化
D[n] ← X[n]
Rec[n] ← n
//动态规划
for i ← n − 1 to 1 do
    if D[i + 1] > 0 then
        D[i] ← X[i] + D[i + 1]
        Rec[i] ← Rec[i + 1]
    end
    else
        D[i] ← X[i]
        Rec[i] ← i
    end
end
```

初始化

动态规划：伪代码

- **Max-Continuous-Subarray-DP(X, n)**

```
输入: 数组X, 数组长度n
输出: 最大子数组和S_max,子数组起止位置l, r
新建一维数组D[1..n]和Rec[1..n]
//初始化
D[n] ← X[n]
Rec[n] ← n
//动态规划
for i ← n − 1 to 1 do
    if D[i + 1] > 0 then
        D[i] ← X[i] + D[i + 1]
        Rec[i] ← Rec[i + 1]
    end
    else
        D[i] ← X[i]
        Rec[i] ← i
    end
end
```

自底向上计算

动态规划：伪代码

- **Max-Continuous-Subarray-DP(X, n)**

输入: 数组X, 数组长度n
输出: 最大子数组和S_{max},子数组起止位置l, r
新建一维数组$D[1..n]$和$Rec[1..n]$
//初始化
$D[n] \leftarrow X[n]$
$Rec[n] \leftarrow n$
//动态规划
for $i \leftarrow n-1$ *to* 1 **do**
  if $D[i+1] > 0$ **then**
    $D[i] \leftarrow X[i] + D[i+1]$
    $Rec[i] \leftarrow Rec[i+1]$
  end
  else
    $D[i] \leftarrow X[i]$
    $Rec[i] \leftarrow i$
  end
end

情况1：$D[i+1] > 0$

动态规划：伪代码

- **Max-Continuous-Subarray-DP(X, n)**

```
输入: 数组X, 数组长度n
输出: 最大子数组和S_max,子数组起止位置l, r
新建一维数组D[1..n]和Rec[1..n]
//初始化
D[n] ← X[n]
Rec[n] ← n
//动态规划
for i ← n − 1 to 1 do
    if D[i + 1] > 0 then
        D[i] ← X[i] + D[i + 1]
        Rec[i] ← Rec[i + 1]
    end
    else
        D[i] ← X[i]
        Rec[i] ← i
    end
end
```

记录子问题结果与决策

动态规划：伪代码

- **Max-Continuous-Subarray-DP(X, n)**

输入: 数组X, 数组长度n
输出: 最大子数组和S_{max},子数组起止位置l, r
新建一维数组$D[1..n]$和$Rec[1..n]$
//初始化
$D[n] \leftarrow X[n]$
$Rec[n] \leftarrow n$
//动态规划
for $i \leftarrow n-1$ *to* 1 **do**
  if $D[i+1] > 0$ **then**
    $D[i] \leftarrow X[i] + D[i+1]$
    $Rec[i] \leftarrow Rec[i+1]$
  end
  else
    $D[i] \leftarrow X[i]$
    $Rec[i] \leftarrow i$
  end
end

情况2： $D[i+1] \leq 0$

动态规划：伪代码

- **Max-Continuous-Subarray-DP(X, n)**

```
//查找解
S_max ← D[1]
for i ← 2 to n do
    if S_max < D[i] then
        S_max ← D[i]
        l ← i
        r ← Rec[i]
    end
end
return S_max, l, r
```

时间复杂度分析

输入: 数组X, 数组长度n
输出: 最大子数组和S_{max},子数组起止位置l, r
新建一维数组$D[1..n]$和$Rec[1..n]$
//初始化
$D[n] \leftarrow X[n]$
$Rec[n] \leftarrow n$
//动态规划
for $i \leftarrow n-1$ *to* 1 **do**
  if $D[i+1] > 0$ **then**
    $D[i] \leftarrow X[i] + D[i+1]$
    $Rec[i] \leftarrow Rec[i+1]$
  end
  else
    $D[i] \leftarrow X[i]$
    $Rec[i] \leftarrow i$
  end
end

$O(n)$

时间复杂度：$O(n)$

谢谢

BDA

Beihang University