Design and Analysis of Algorithms

Lecture 8: Selection Problem

童咏昕

北京航空航天大学
计算机学院

问题背景：最小值查找

- 给定数组$A[1..16]$，寻找其中最小值

21	17	37	28	13	14	22	52	40	24	48	4	47	8	42	18

问题背景：最小值查找

- 给定数组$A[1..16]$，寻找其中最小值

21	17	37	28	13	14	22	52	40	24	48	4	47	8	42	18

- 依次扫描，记录最小值

21	17	37	28	13	14	22	52	40	24	48	4	47	8	42	18

扫描

问题背景：最小值查找

- 给定数组$A[1..16]$，寻找其中最小值

21	17	37	28	13	14	22	52	40	24	48	4	47	8	42	18

- 依次扫描，记录最小值

21	17	37	28	13	14	22	52	40	24	48	4	47	8	42	18

扫描

问题：如何求得数组中第k小的元素？

问题定义

- 形式化定义

次序选择问题

Selection Problem

输入

- 包含n个不同元素的数组$A[1..n]$
- 整数$k(1 \le k \le n)$

问题定义

- 形式化定义

次序选择问题

Selection Problem

输入

- 包含n个不同元素的数组$A[1..n]$
- 整数$k(1 \le k \le n)$

输出

- 数组$A[1..n]$中第k小的元素$(1 \le k \le n)$

排序求解

- 数组排序
  - 求得所有元素的次序
  - 时间复杂度：$O(n \log n)$

21	17	37	28	13	14	22	52	40	24	48	4	47	8	42	18

↓ 排序

4	8	13	14	17	18	21	22	24	28	37	40	42	47	48	52

排序求解

- 数组排序
  - 求得所有元素的次序
  - 时间复杂度：$O(n\log n)$
- 选择元素
  - 求得**第8小**的元素
  - 时间复杂度：$O(1)$

21	17	37	28	13	14	22	52	40	24	48	4	47	8	42	18

排序

查找第8小元素

4	8	13	14	17	18	21	22	24	28	37	40	42	47	48	52

问题分析

- 数组排序
  - 求得所有元素的次序
  - 时间复杂度：$O(n\log n)$

第6小元素 第8小元素 第15小元素

4	8	13	14	17	18	21	22	24	28	37	40	42	47	48	52

问题分析

- 数组排序
  - 求得所有元素的次序
  - 时间复杂度：$O(n\log n)$

					第6小元素		第8小元素							第15小元素	
4	8	13	14	17	18	21	22	24	28	37	40	42	47	48	52

问题：是否有必要求得所有元素的次序？

问题分析

- 次序选择
  - 不必求得所有元素次序
  - 时间复杂度：$O(?)$

第6小元素 第8小元素 第15小元素

4	8	13	14	17	18	21	22	24	28	37	40	42	47	48	52

查找第8小元素

17	13	14	4	8	18	21	22	28	24	37	40	47	52	42	48

7个元素

问题分析

- 次序选择
  - 不必求得所有元素次序
  - 时间复杂度：$O(?)$

第6小元素 第8小元素 第15小元素

4	8	13	14	17	18	21	22	24	28	37	40	42	47	48	52

查找第8小元素

17	13	14	4	8	18	21	22	28	24	37	40	47	52	42	48

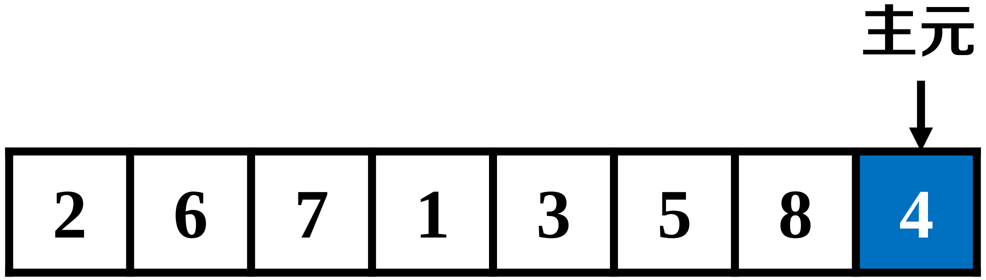
受启发于快速排序的数组划分

回顾与启发

2	6	7	1	3	5	8	4

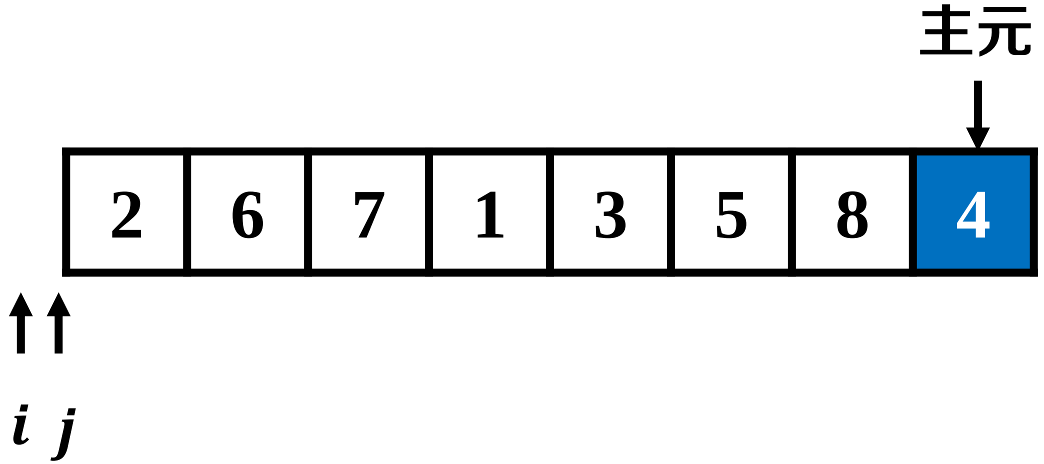
回顾与启发

- 选取固定位置主元x（如尾元素）



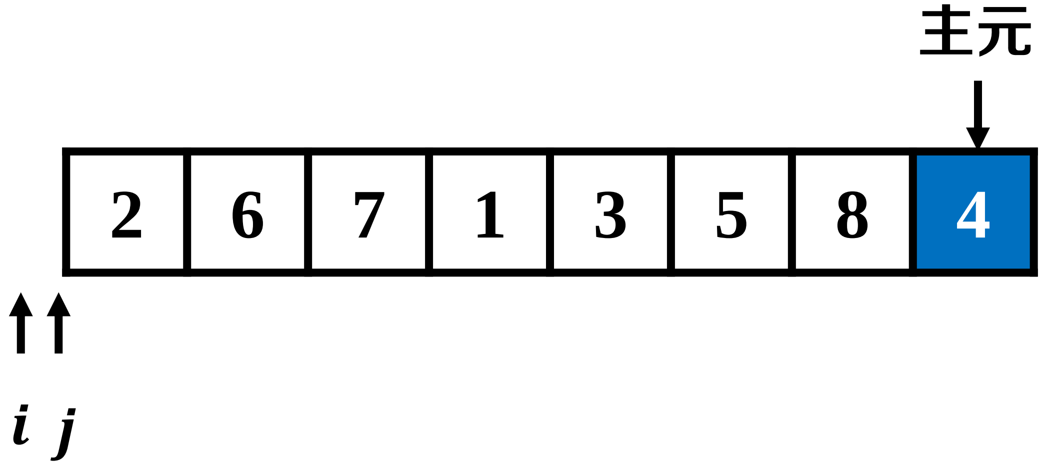
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j



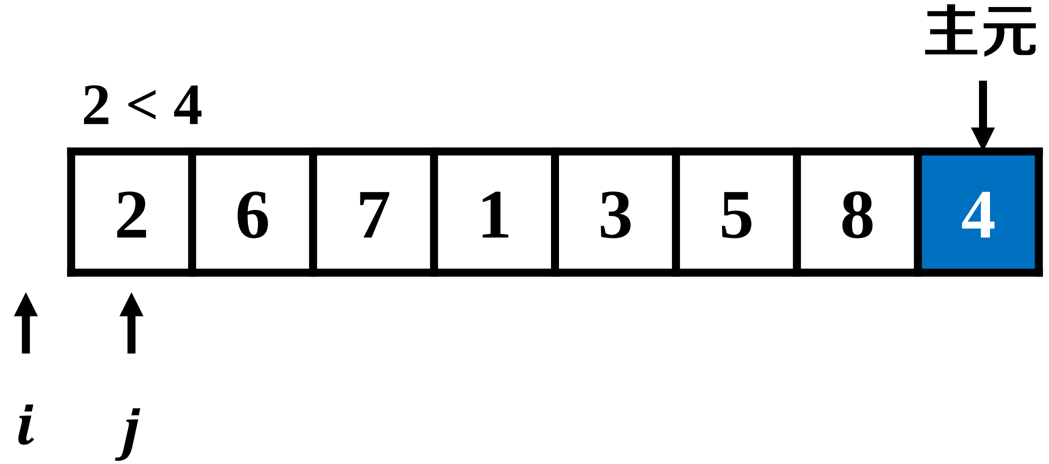
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \le x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



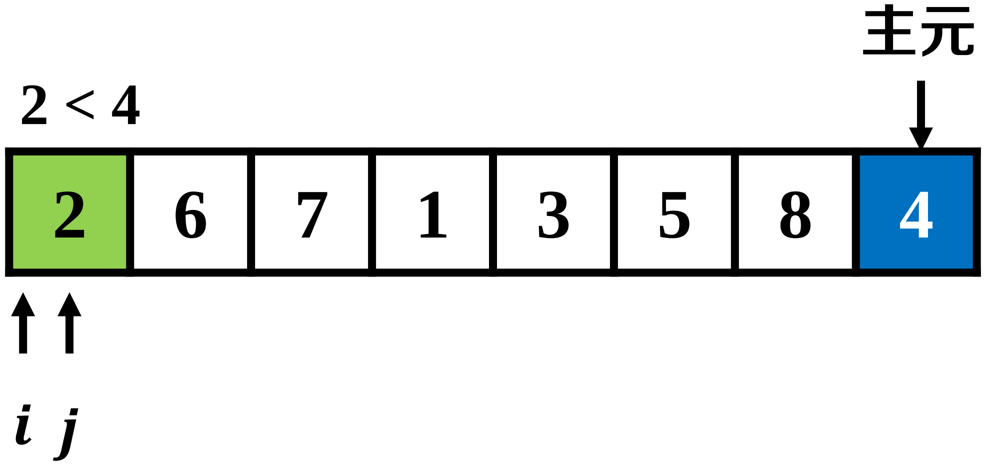
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \le x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



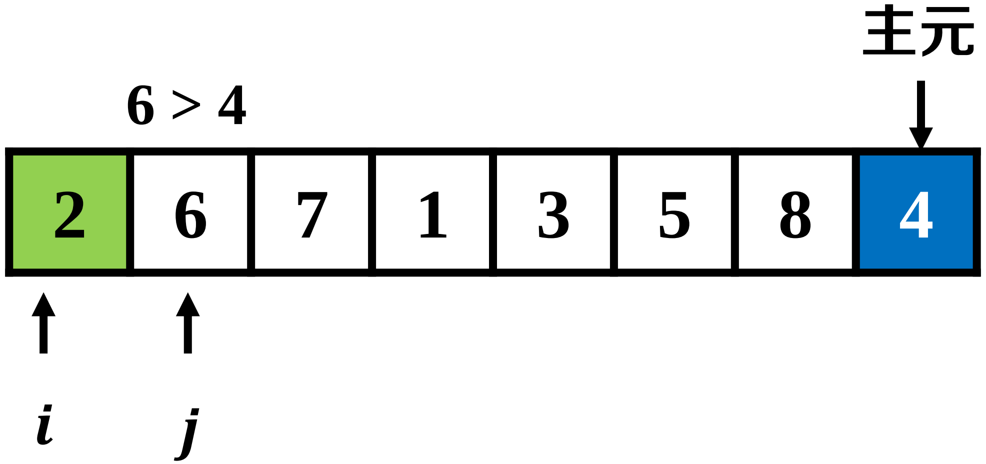
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \leq x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



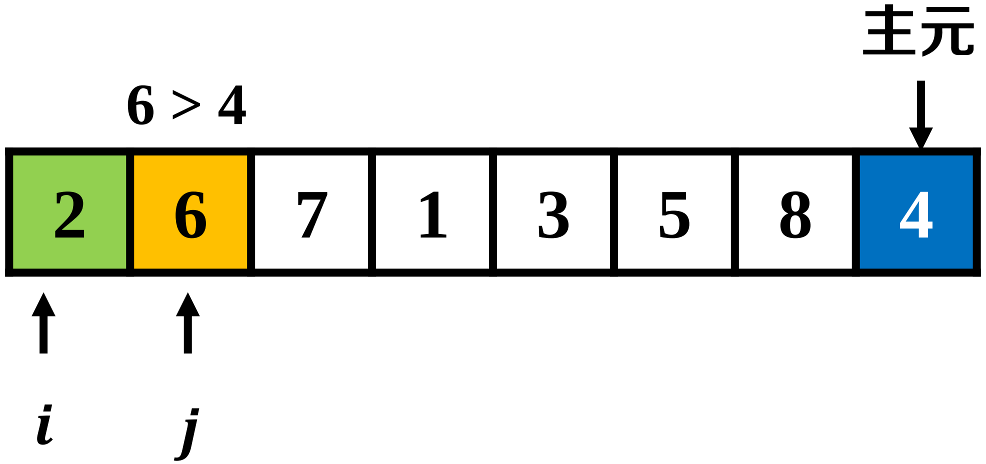
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \leq x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



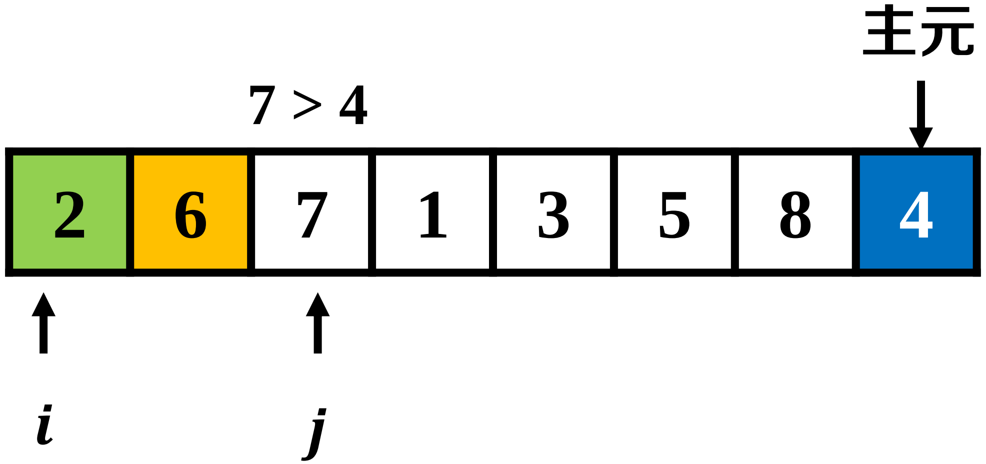
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \le x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



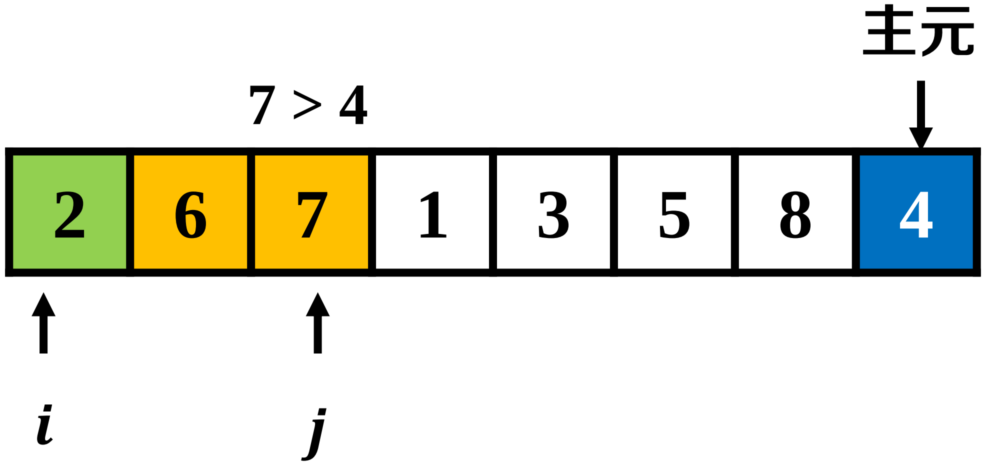
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \le x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



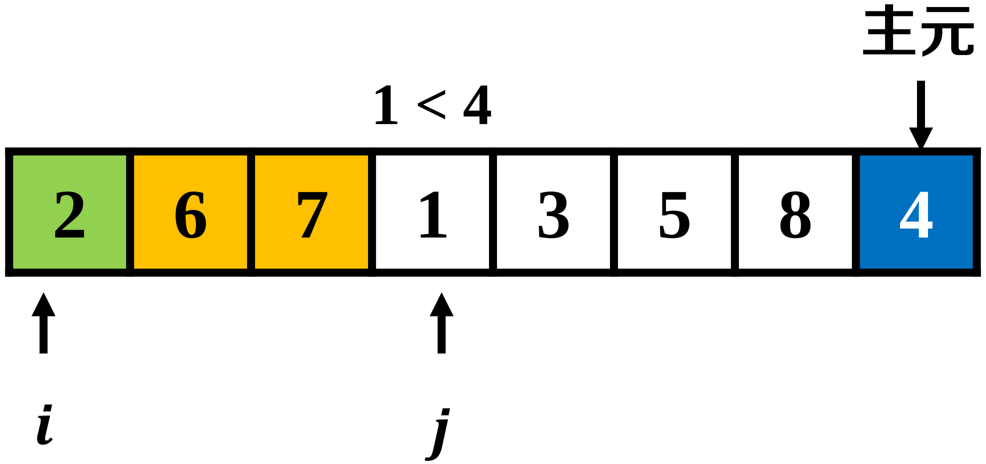
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \leq x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



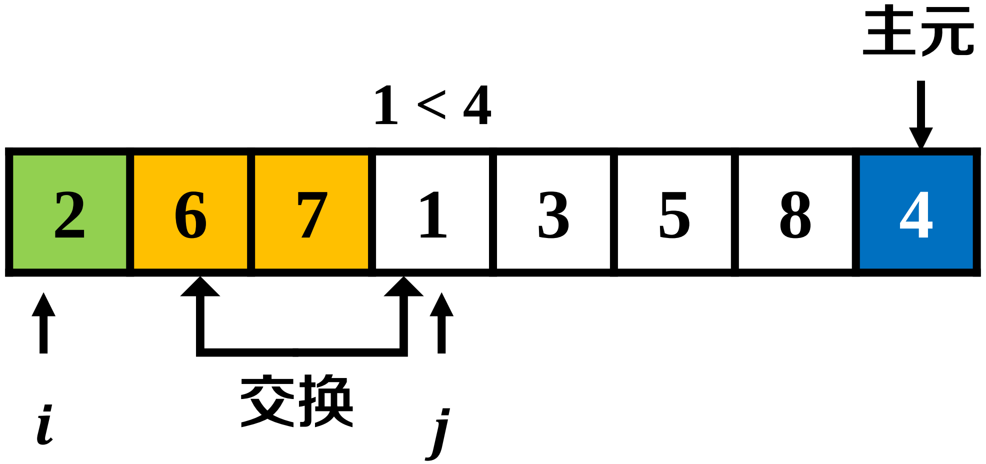
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \leq x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



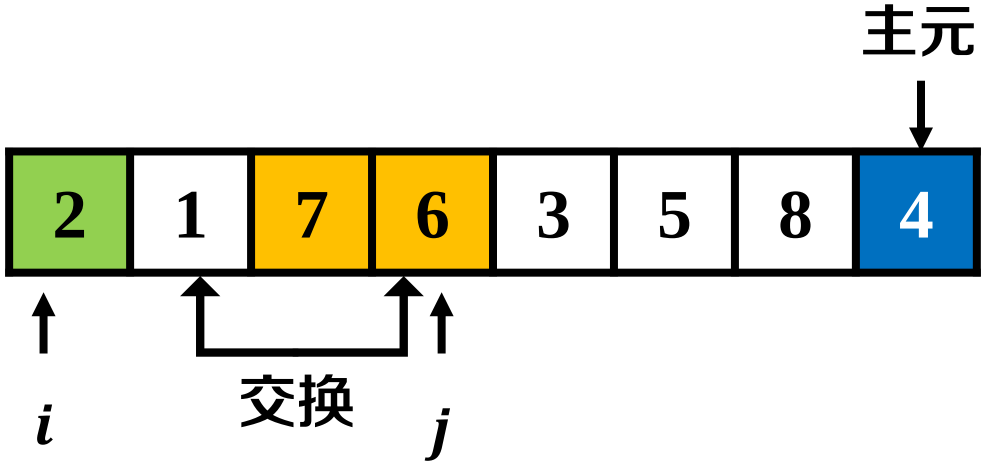
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \leq x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



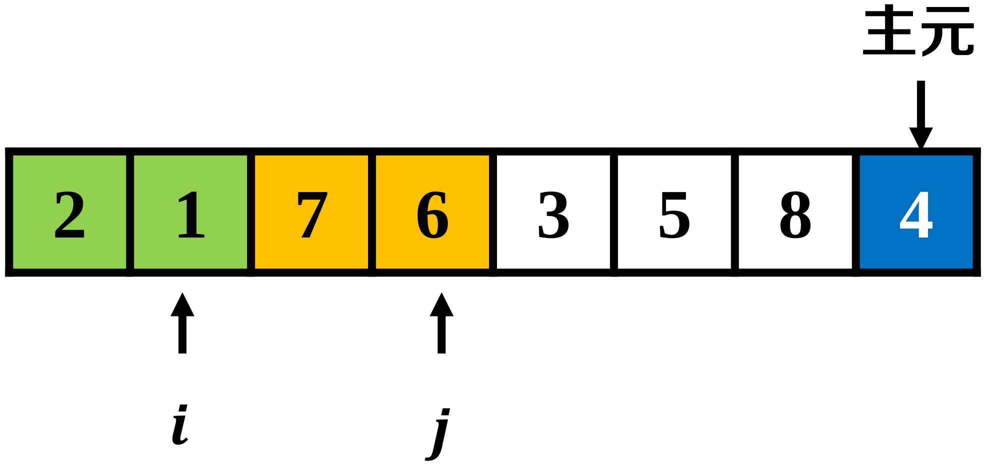
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \le x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



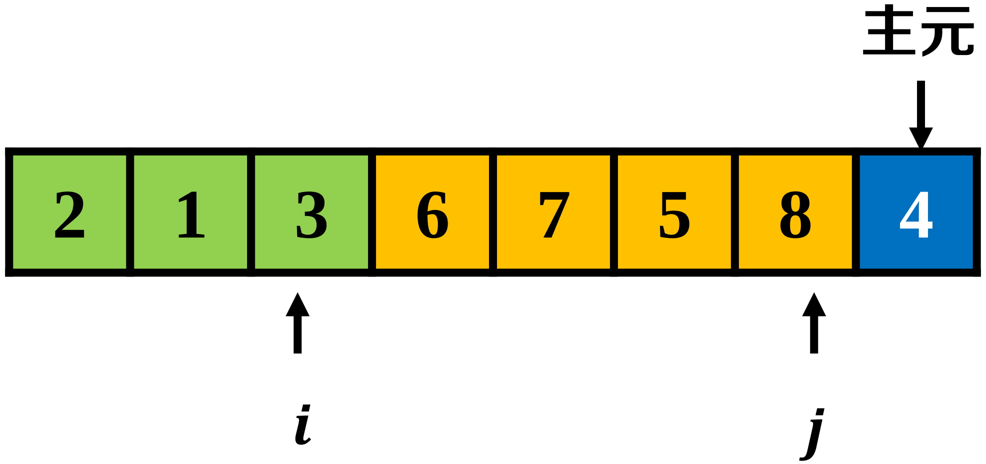
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \leq x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



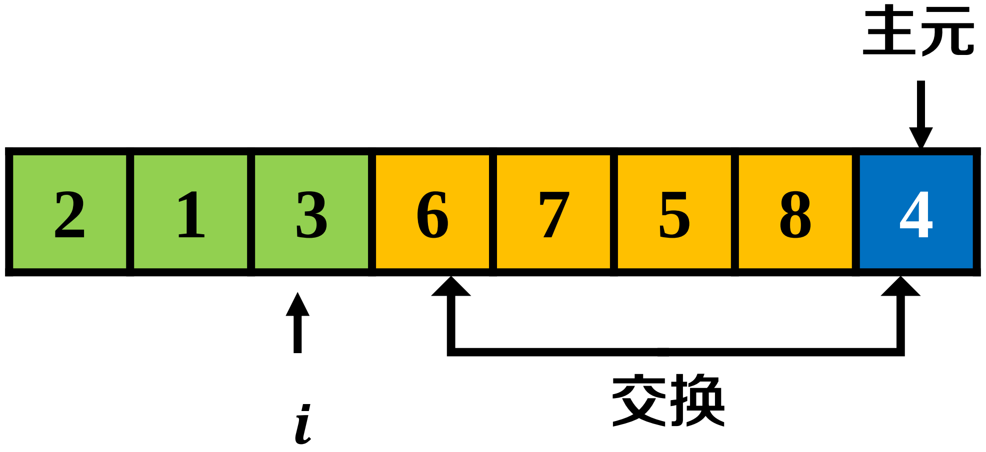
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \leq x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



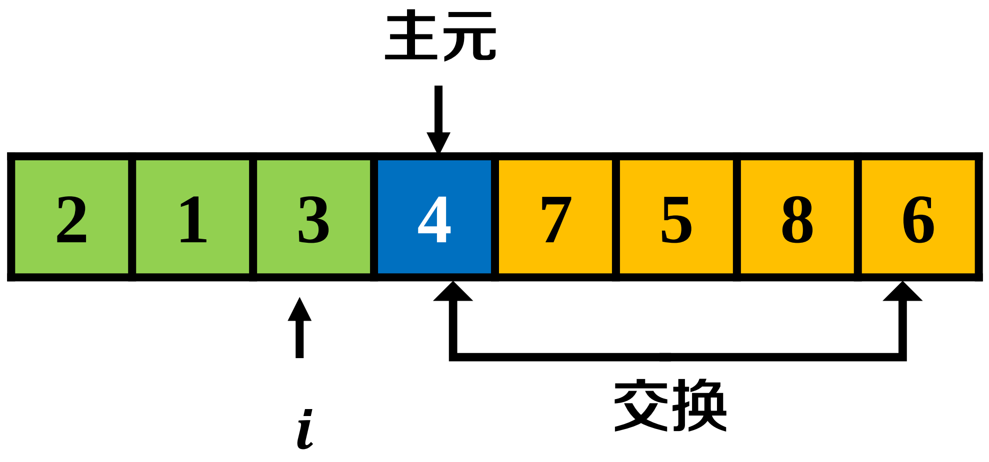
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \le x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



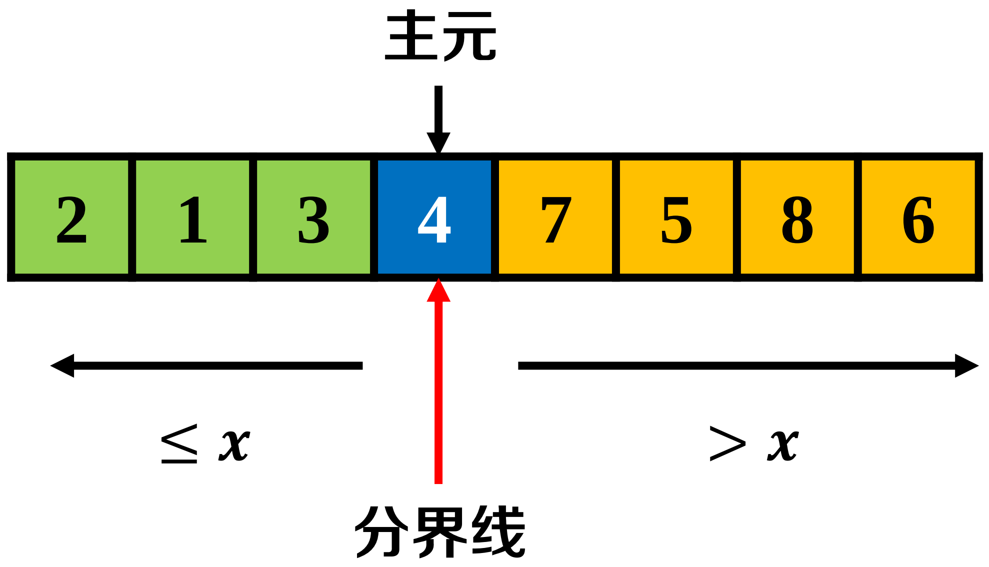
回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \le x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



回顾与启发

- 选取固定位置主元x（如尾元素）
- 维护两个部分的右端点变量i, j
- 考察数组元素$A[j]$，只和主元比较
  - 若$A[j] \leq x$，则交换$A[j]$和$A[i+1]$，i与j右移
  - 若$A[j] > x$，则j右移



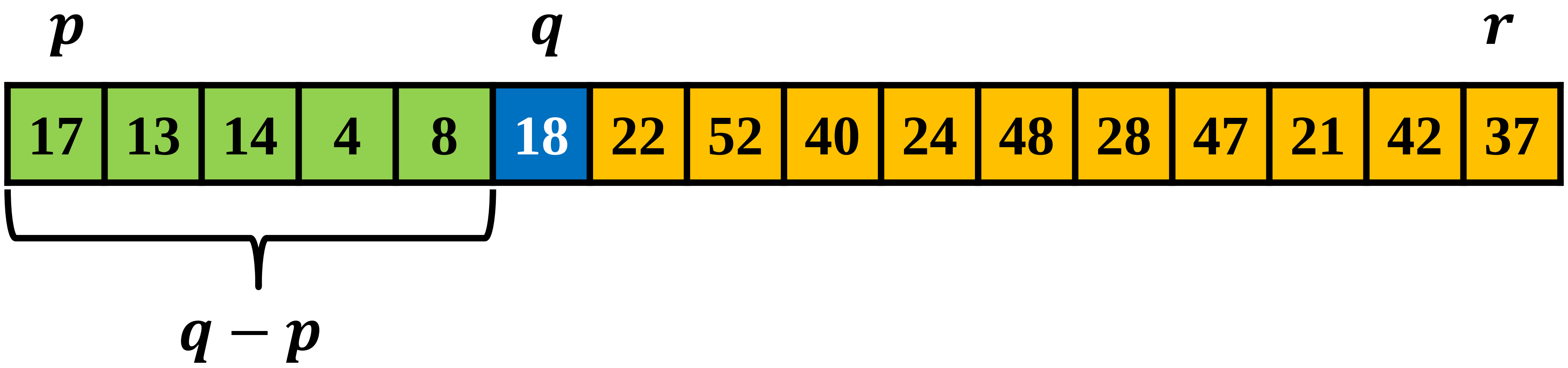
固定位置划分求解

- 选取固定位置主元

p															r
21	17	37	28	13	14	22	52	40	24	48	4	47	8	42	18
															主元

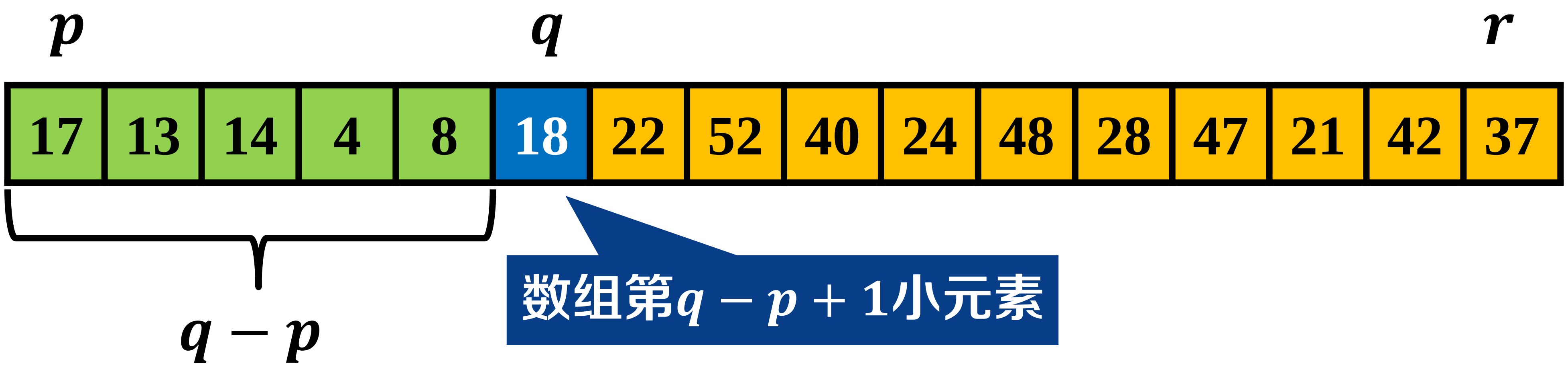
固定位置划分求解

- 选取固定位置主元，小于主元的元素个数$q-p$



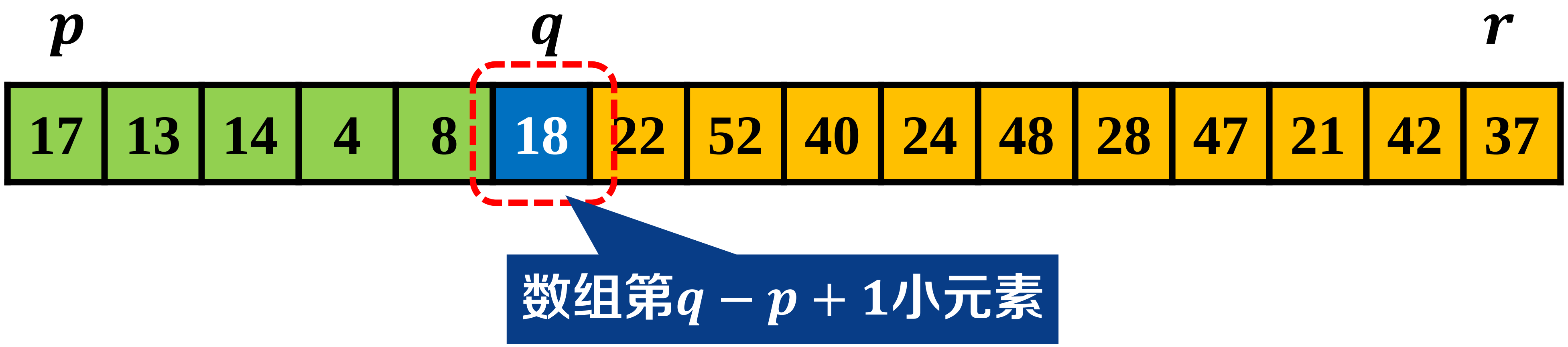
固定位置划分求解

- 选取固定位置主元，小于主元的元素个数$q-p$



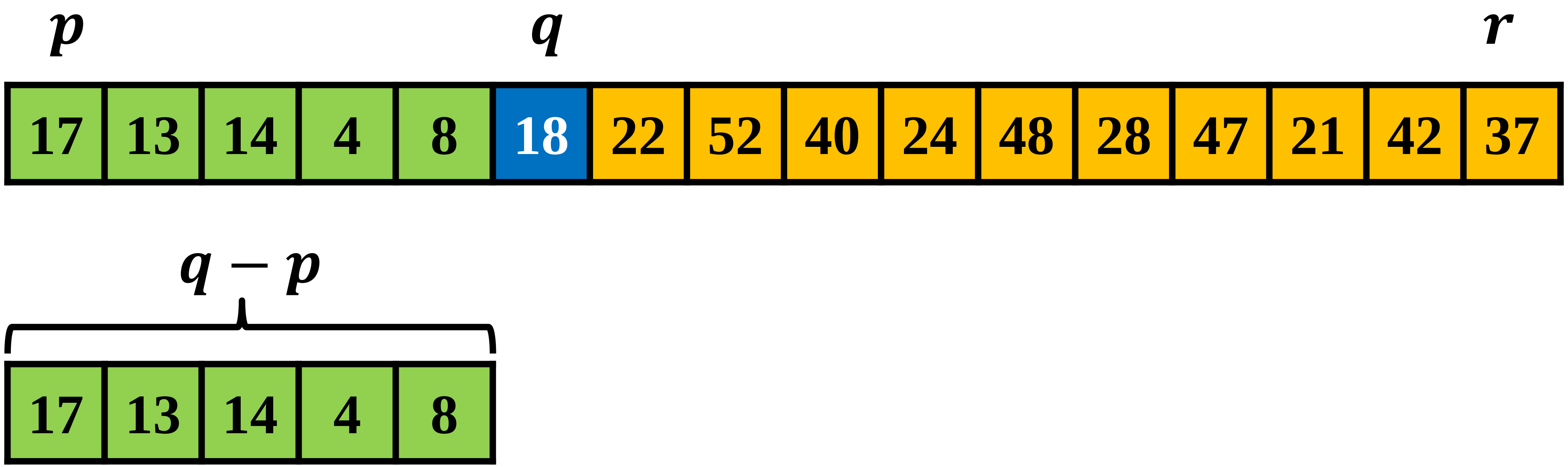
固定位置划分求解

- 选取固定位置主元，小于主元的元素个数$q-p$
  - 情况1：$k=q-p+1$，$A[q]$为数组第k小元素



固定位置划分求解

- 选取固定位置主元，小于主元的元素个数$q-p$
  - 情况1：$k=q-p+1$，$A[q]$为数组第k小元素
  - 情况2：$k<q-p+1$，数组第k小元素在$A[p..q-1]$中



固定位置划分求解

- 选取固定位置主元，小于主元的元素个数$q-p$
  - 情况1：$k=q-p+1$，$A[q]$为数组第k小元素
  - 情况2：$k<q-p+1$，在$A[p..q-1]$中寻找第k小元素

p					q										r
17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

$$\overbrace{17\quad 13\quad 14}^{k}\quad 4\quad 8$$

固定位置划分求解

- 选取固定位置主元，小于主元的元素个数$q-p$
  - 情况1：$k=q-p+1$，$A[q]$为数组第k小元素
  - 情况2：$k<q-p+1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k>q-p+1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

p q r

17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

$$\overbrace{17\ \ 13\ \ 14\ \ 4\ \ 8\ \ 18}^{q-p+1}\ \ \overbrace{22\ \ 52\ \ 40\ \ 24\ \ 48}^{k-(q-p+1)}\ \ 28\ \ 47\ \ 21\ \ 42\ \ 37$$

固定位置划分求解

原问题解在某个子问题中

分而治之框架

17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

分解原问题

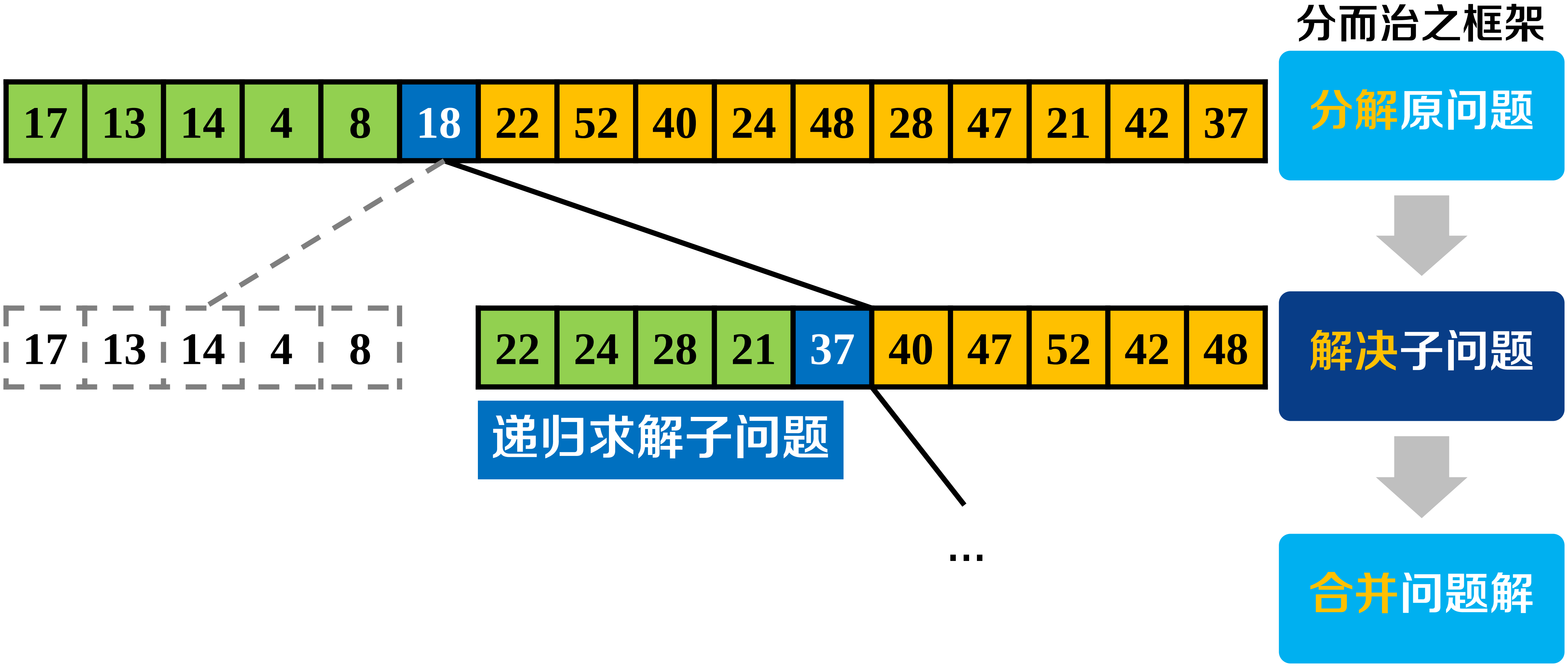
17	13	14	4	8

22	52	40	24	48	28	47	21	42	37

解决子问题

合并问题解

固定位置划分求解


分而治之框架
分解原问题
解决子问题
合并问题解
17 13 14 4 8 18 22 52 40 24 48 28 47 21 42 37
17 13 14 4 8
22 24 28 21 37 40 47 52 42 48
递归求解子问题
…

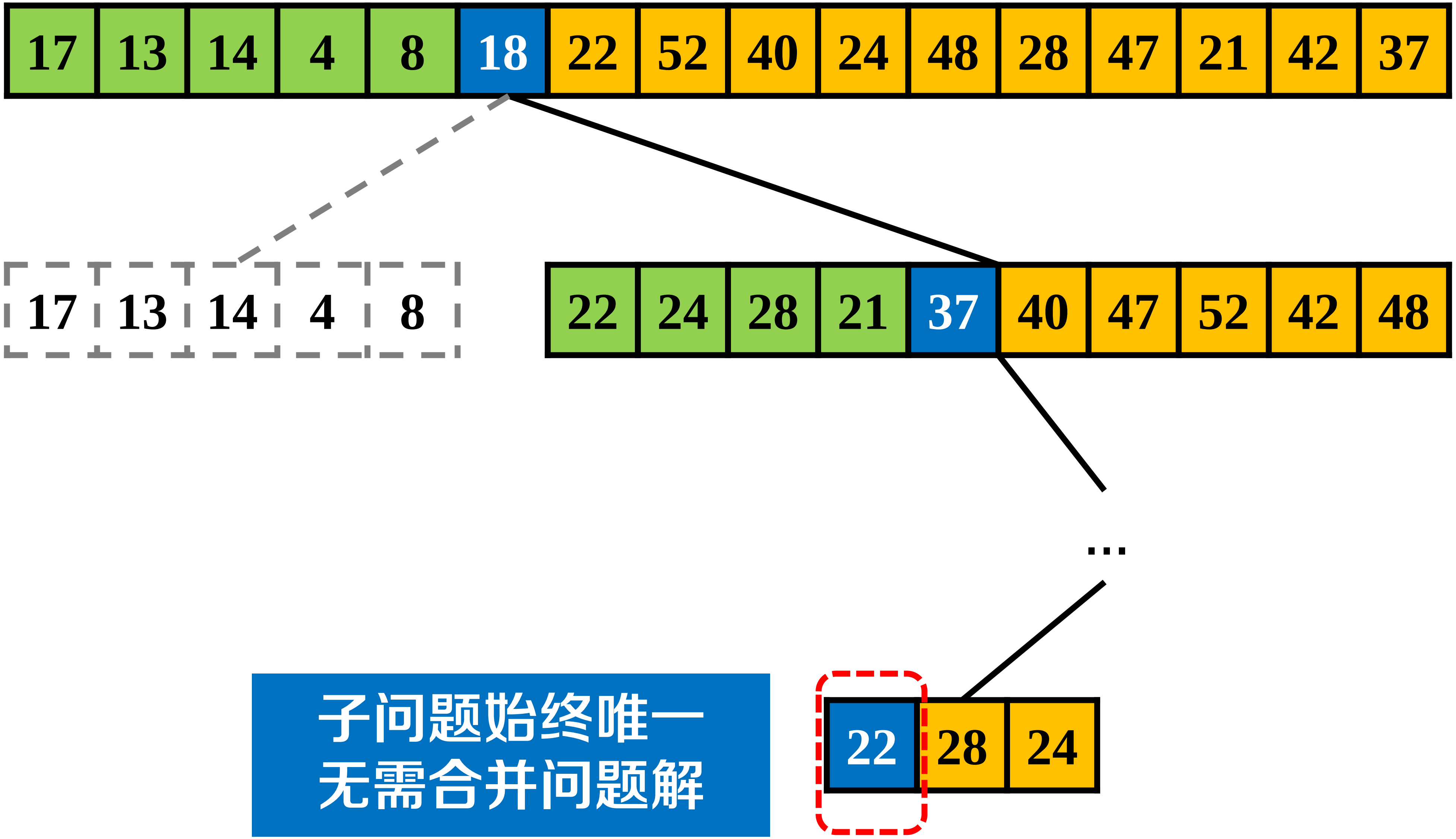
固定位置划分求解

分而治之框架

分解原问题

解决子问题

合并问题解



子问题始终唯一
无需合并问题解

算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

$p = 1$

21	17	37	28	13	14	22	52	40	24	48	4	47	8	42	18

$r = 16$

查找第8小元素

算法实例

- **选取固定位置主元**
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

$p = 1$

21	17	37	28	13	14	22	52	40	24	48	4	47	8	42	18

$r = 16$

主元

查找第8小元素

算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

$p = 1$

17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

$r = 16$

主元

查找第8小元素 $q = 6$

算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

$p = 1$

17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

$r = 16$

主元

查找第8小元素 $q = 6$

$$q - p + 1 = 6 - 1 + 1 = 6$$

算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

$p = 1$	17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37	$r = 16$

主元

查找第8小元素 $q = 6$

$$q - p + 1 = 6 - 1 + 1 = 6$$

算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

$p = 7$

22	52	40	24	48	28	47	21	42	37

$r = 16$

查找第2小元素

算法实例

- **选取固定位置主元**
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

$p = 7$

22	52	40	24	48	28	47	21	42	37

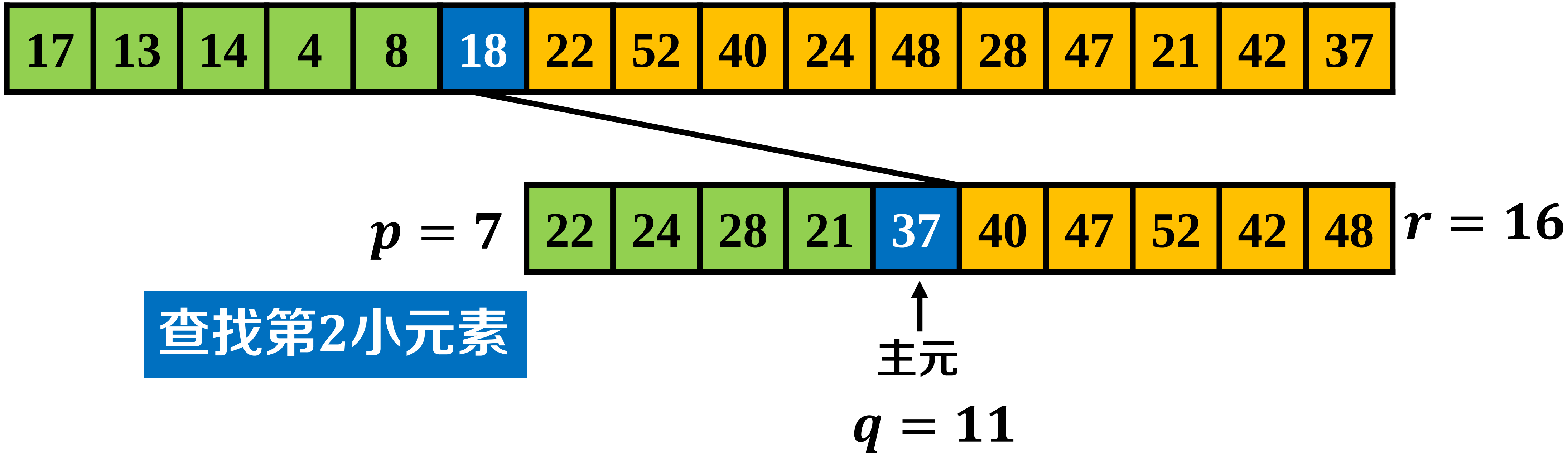
$r = 16$

查找第2小元素

主元

算法实例

- 选取固定位置主元
  - 情况1：$k=q-p+1$，$A[q]$为数组第k小元素
  - 情况2：$k<q-p+1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k>q-p+1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素



查找第2小元素

算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k - (q - p + 1)$小元素

17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

$p = 7$

22	24	28	21	37	40	47	52	42	48

$r = 16$

↑
主元

$q = 11$

查找第2小元素

$q - p + 1 = 11 - 7 + 1 = 5$

算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

$p = 7$

22	24	28	21	37	40	47	52	42	48

$r = 16$

主元

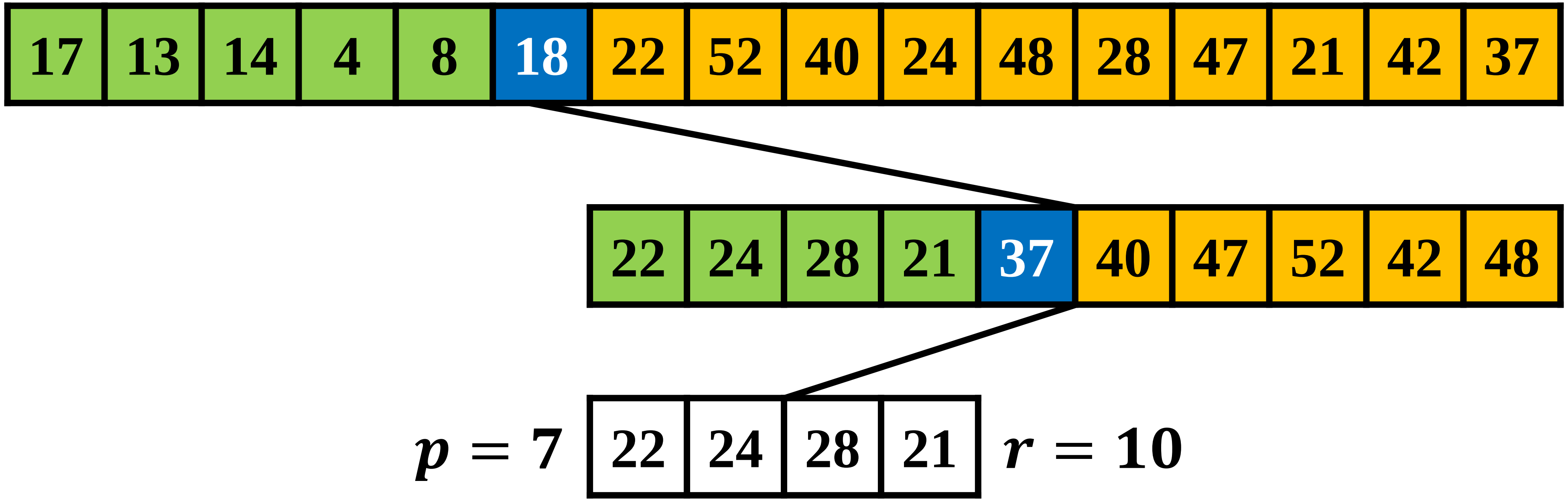
$q = 11$

查找第2小元素

$q - p + 1 = 11 - 7 + 1 = 5$

算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素



查找第2小元素

算法实例

- **选取固定位置主元**
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

22	24	28	21	37	40	47	52	42	48

$p = 7$

22	24	28	21

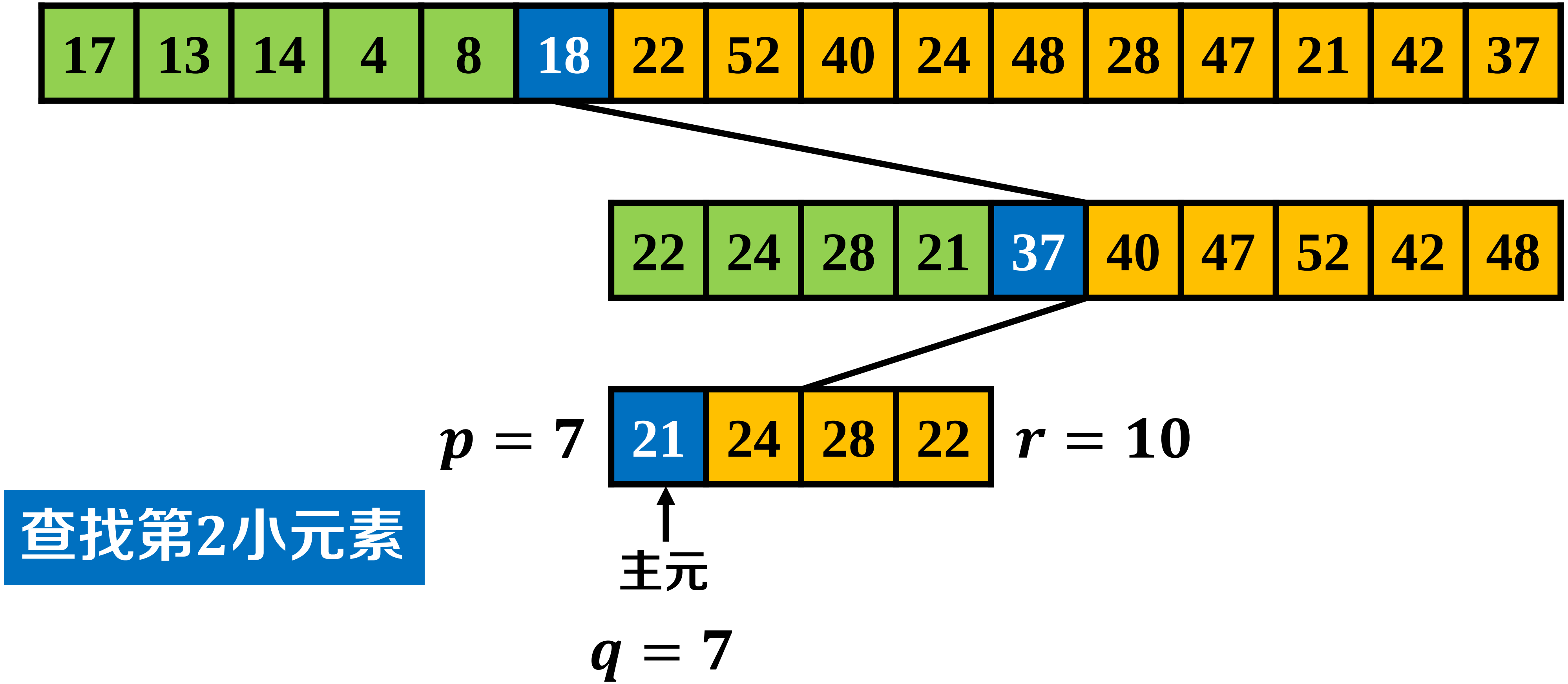
$r = 10$

主元

查找第2小元素

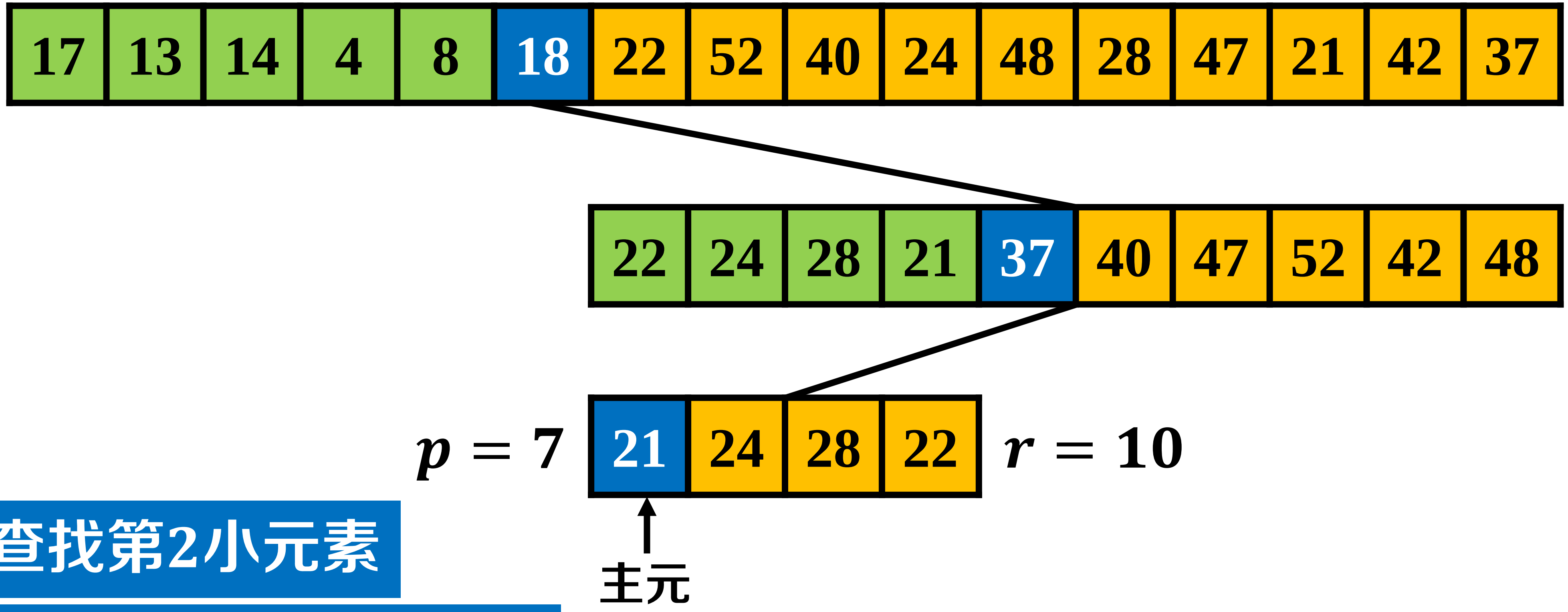
算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素



算法实例

- **选取固定位置主元**
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素



查找第2小元素

$q - p + 1 = 7 - 7 + 1 = 1$

$q = 7$

算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

22	24	28	21	37	40	47	52	42	48

$p = 7$

21	24	28	22

$r = 10$

主元

查找第2小元素

$q - p + 1 = 7 - 7 + 1 = 1$

$q = 7$

算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素

17	13	14	4	8	18	22	52	40	24	48	28	47	21	42	37

22	24	28	21	37	40	47	52	42	48

21	24	28	22

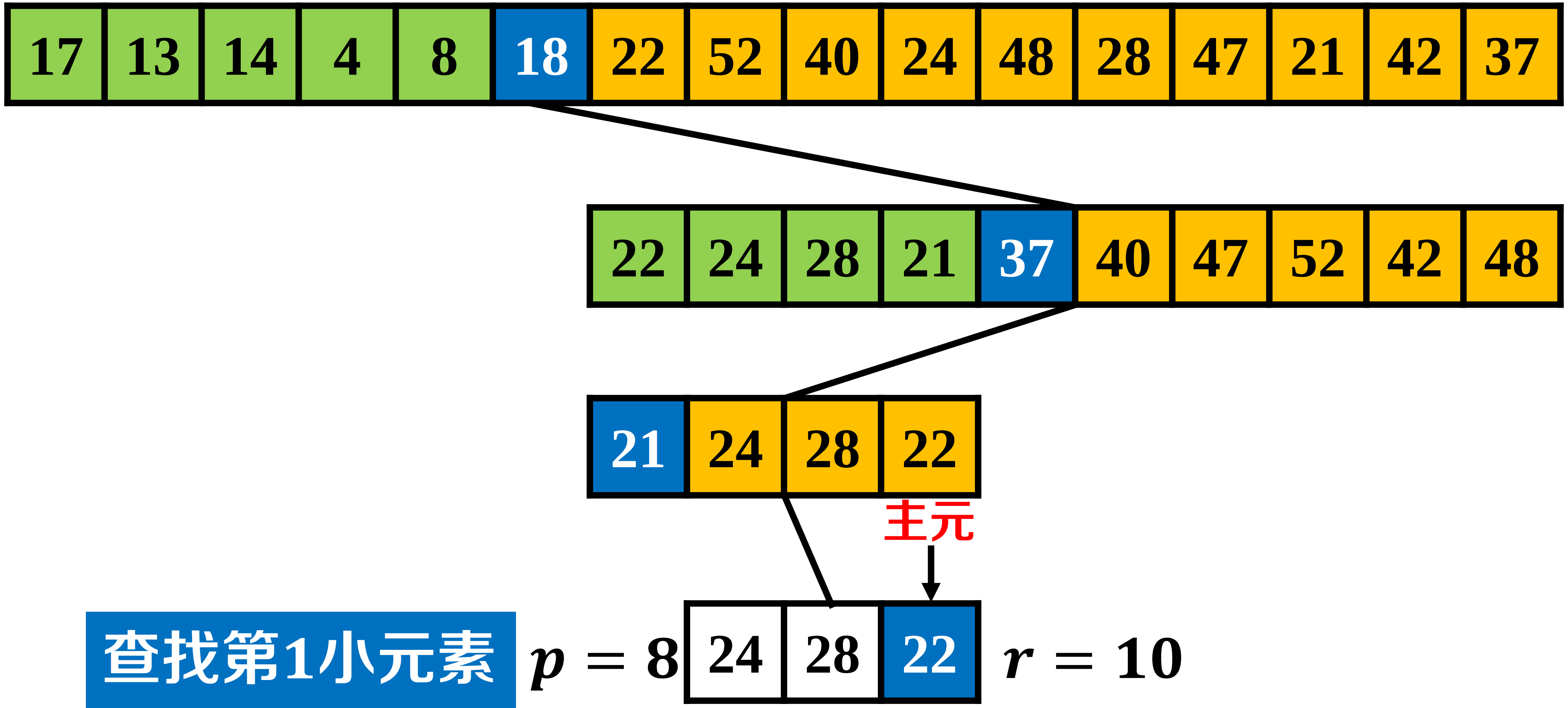
查找第1小元素 $p = 8$

24	28	22

$r = 10$

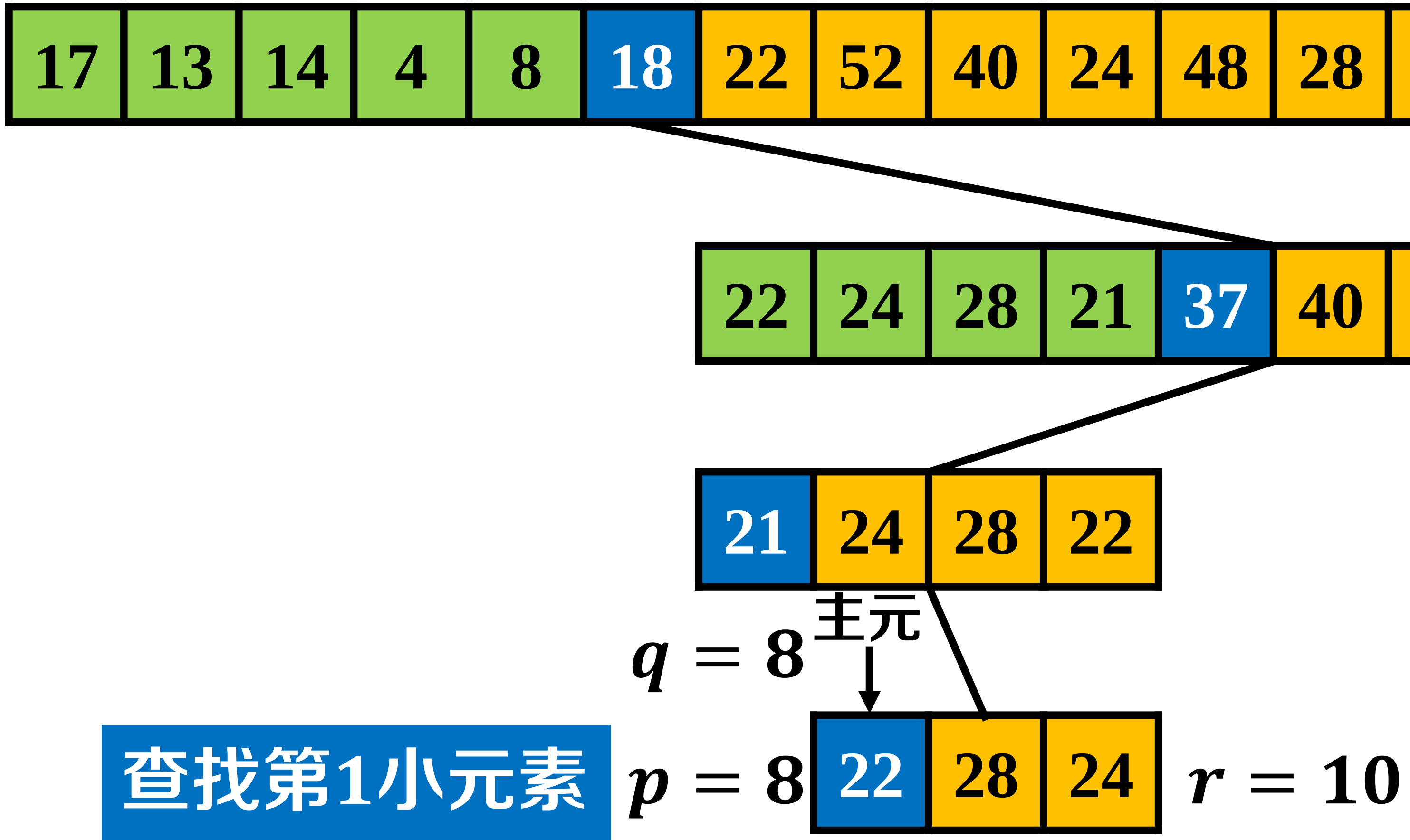
算法实例

- **选取固定位置主元**
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素



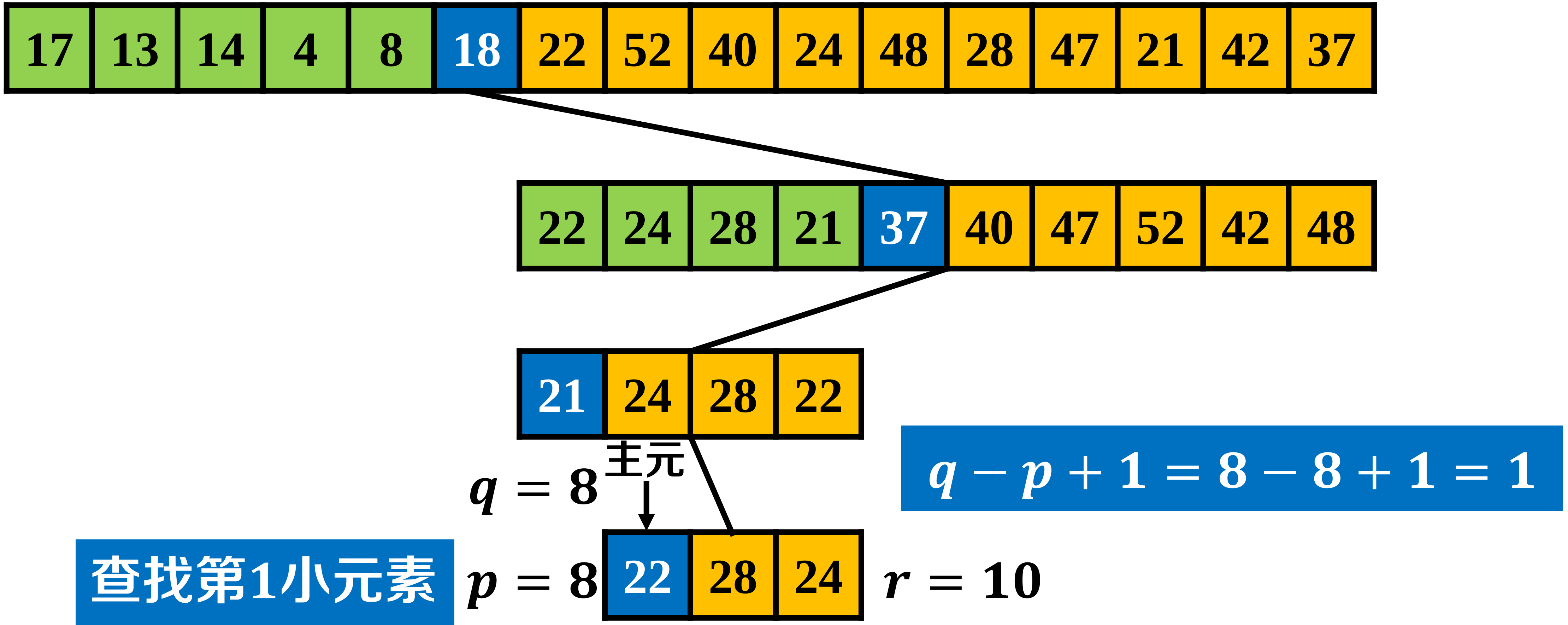
算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素



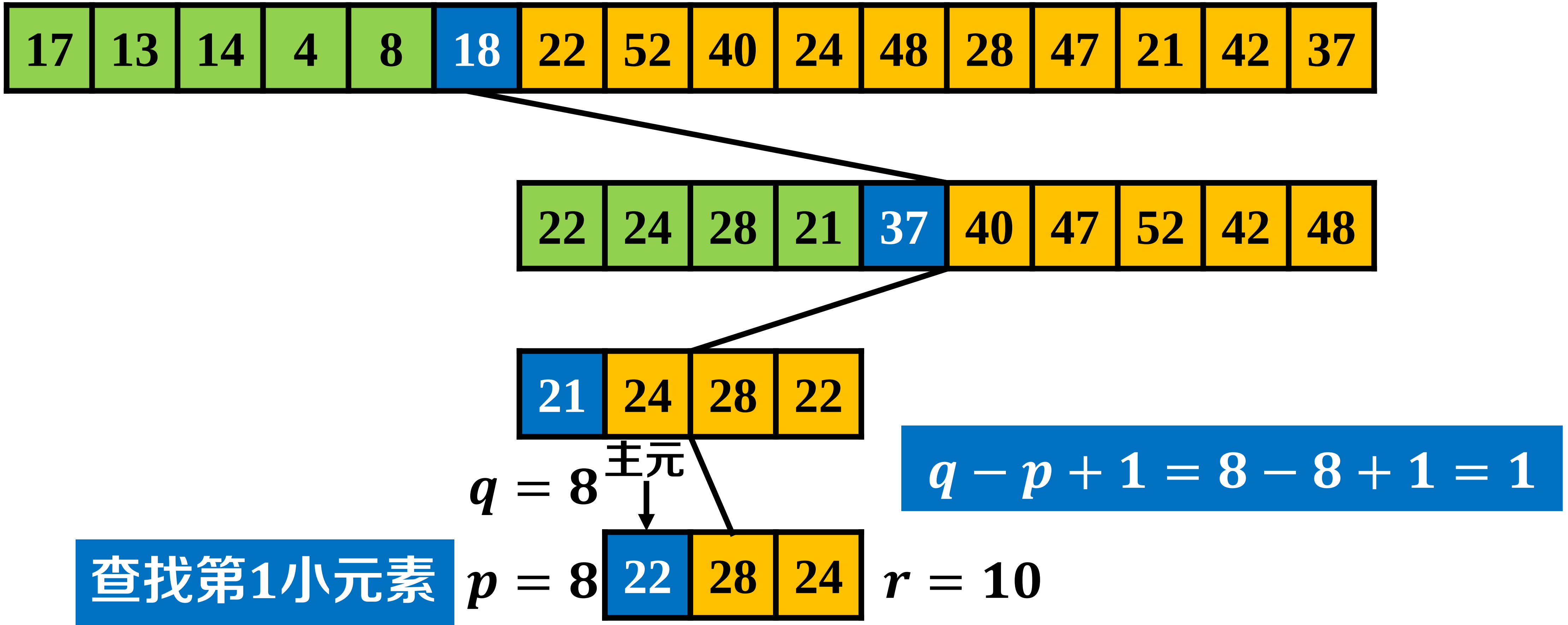
算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素



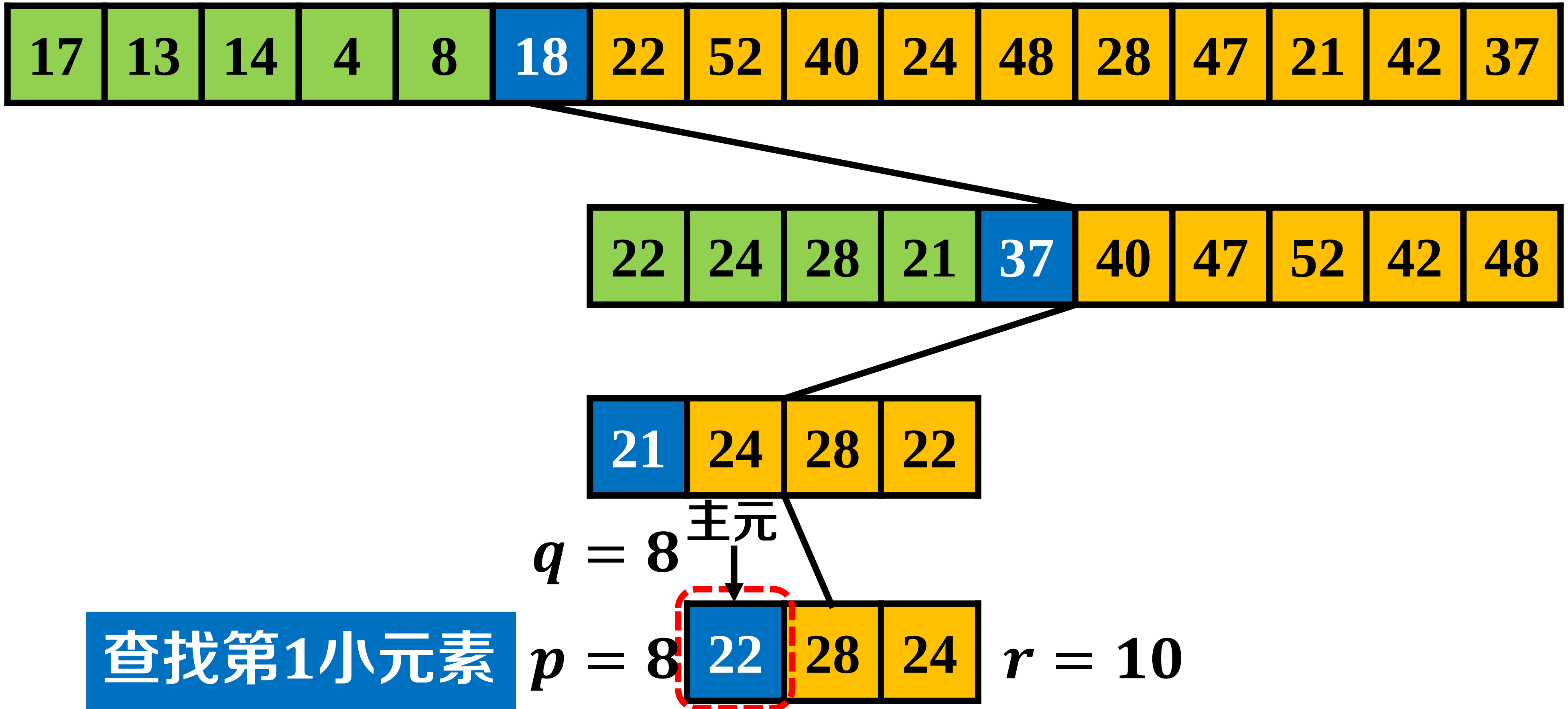
算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素



算法实例

- 选取固定位置主元
  - 情况1：$k = q - p + 1$，$A[q]$为数组第k小元素
  - 情况2：$k < q - p + 1$，在$A[p..q-1]$中寻找第k小元素
  - 情况3：$k > q - p + 1$，在$A[q+1..r]$中寻找第$k-(q-p+1)$小元素



固定位置划分：伪代码

- **Partition(A, p, r)**

输入: 数组A, 起始位置p, 终止位置r
输出: 划分位置q

$x \leftarrow A[r]$ 　　选取主元
$i \leftarrow p - 1$
for $j \leftarrow p$ *to* $r - 1$ **do**
　　if $A[j] \leq x$ **then**
　　　　exchange $A[i+1]$ with $A[j]$
　　　　$i \leftarrow i + 1$
　　end
end
exchange $A[i+1]$ with $A[r]$
$q \leftarrow i + 1$
return q

固定位置划分：伪代码

- **Partition(A, p, r)**

输入: 数组A, 起始位置p, 终止位置r
输出: 划分位置q

$x \leftarrow A[r]$
$i \leftarrow p - 1$
for $j \leftarrow p$ *to* $r - 1$ **do**
  if $A[j] \leq x$ **then**
    exchange $A[i + 1]$ with $A[j]$
    $i \leftarrow i + 1$
  end
end
exchange $A[i + 1]$ with $A[r]$
$q \leftarrow i + 1$
return q

初始化下标

固定位置划分：伪代码

- **Partition(A, p, r)**

输入: 数组A, 起始位置p, 终止位置r
输出: 划分位置q

$x \leftarrow A[r]$
$i \leftarrow p - 1$
for $j \leftarrow p$ *to* $r - 1$ **do**
  if $A[j] \leq x$ **then**
    exchange $A[i+1]$ with $A[j]$
    $i \leftarrow i + 1$
  end
end
exchange $A[i+1]$ with $A[r]$
$q \leftarrow i + 1$
return q

依次扫描

固定位置划分：伪代码

- **Partition(A, p, r)**

输入: 数组A, 起始位置p, 终止位置r
输出: 划分位置q

$x \leftarrow A[r]$
$i \leftarrow p - 1$
for $j \leftarrow p$ *to* $r - 1$ **do**
  if $A[j] \leq x$ **then**
    exchange $A[i+1]$ with $A[j]$
    $i \leftarrow i + 1$
  end
end
exchange $A[i+1]$ with $A[r]$
$q \leftarrow i + 1$
return q

比主元小的元素交换到前面

固定位置划分：伪代码

- **Partition(A, p, r)**

```
输入: 数组A, 起始位置p, 终止位置r
输出: 划分位置q
x ← A[r]
i ← p − 1
for j ← p to r − 1 do
    if A[j] ≤ x then
        exchange A[i + 1] with A[j]
        i ← i + 1
    end
end
exchange A[i + 1] with A[r]
q ← i + 1
return q
```

主元作分界线

固定位置划分：伪代码

- **Partition(A, p, r)**

输入: 数组A, 起始位置p, 终止位置r
输出: 划分位置q

$x \leftarrow A[r]$
$i \leftarrow p - 1$
for $j \leftarrow p$ *to* $r - 1$ **do**
  if $A[j] \leq x$ **then**
    exchange $A[i+1]$ with $A[j]$
    $i \leftarrow i + 1$
  end
end
exchange $A[i+1]$ with $A[r]$
$q \leftarrow i + 1$
return q

返回划分位置

固定位置次序选择：伪代码

- Selection(A, p, r, k)

输入: 数组A,起始位置p,终止位置r,元素次序k
输出: 第k小元素x

$q \leftarrow$ Partition(A, p, r)

划分数组

if $k = (q - p + 1)$ **then**
  $x \leftarrow A[q]$
end
if $k < (q - p + 1)$ **then**
  $x \leftarrow$ Selection$(A, p, q - 1, k)$
end
if $k > (q - p + 1)$ **then**
  $x \leftarrow$ Selection$(A, q + 1, r, k - (q - p + 1))$
end
return x

固定位置次序选择：伪代码

- **Selection(A, p, r, k)**

输入: 数组A,起始位置p,终止位置r,元素次序k
输出: 第k小元素x

$q \leftarrow \text{Partition(A, p, r)}$
if $k = (q-p+1)$ **then**
| $x \leftarrow A[q]$
end
if $k < (q-p+1)$ **then**
| $x \leftarrow \text{Selection}(A, p, q-1, k)$
end
if $k > (q-p+1)$ **then**
| $x \leftarrow \text{Selection}(A, q+1, r, k-(q-p+1))$
end
return x

主元为第k小元素

固定位置次序选择：伪代码

- **Selection(A, p, r, k)**

输入: 数组A,起始位置p,终止位置r,元素次序k
输出: 第k小元素x

$q \leftarrow \text{Partition(A, p, r)}$
if $k = (q - p + 1)$ **then**
| $x \leftarrow A[q]$
end
if $k < (q - p + 1)$ **then**
| $x \leftarrow \text{Selection}(A, p, q - 1, k)$
end
if $k > (q - p + 1)$ **then**
| $x \leftarrow \text{Selection}(A, q + 1, r, k - (q - p + 1))$
end
return x

第k小元素在$A[p..q-1]$中

固定位置次序选择：伪代码

- **Selection(A, p, r, k)**

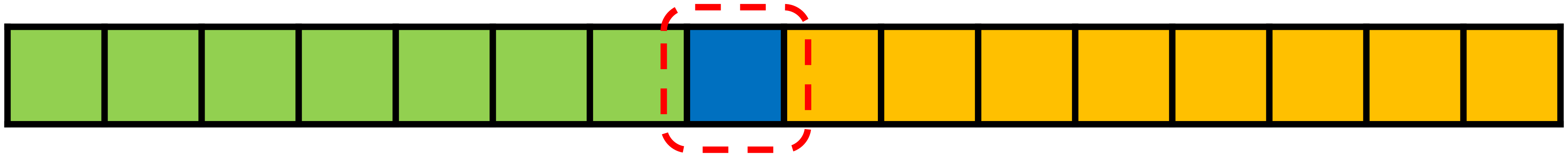
输入: 数组A,起始位置p,终止位置r,元素次序k
输出: 第k小元素x

$q \leftarrow \text{Partition(A, p, r)}$
if $k = (q - p + 1)$ **then**
| $x \leftarrow A[q]$
end
if $k < (q - p + 1)$ **then**
| $x \leftarrow \text{Selection}(A, p, q - 1, k)$
end
if $k > (q - p + 1)$ **then**
| $x \leftarrow \text{Selection}(A, q + 1, r, k - (q - p + 1))$
end
return x

第k小元素在$A[q+1..r]$中

复杂度分析：最好情况

$O(n)$

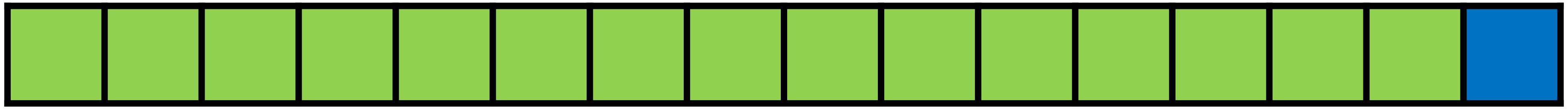


第k小元素

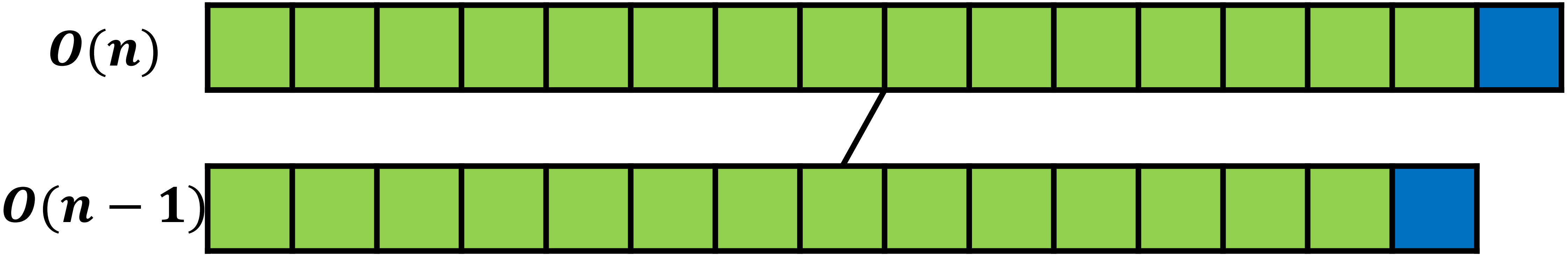
- 时间复杂度：$T(n) = O(n)$

复杂度分析：最坏情况

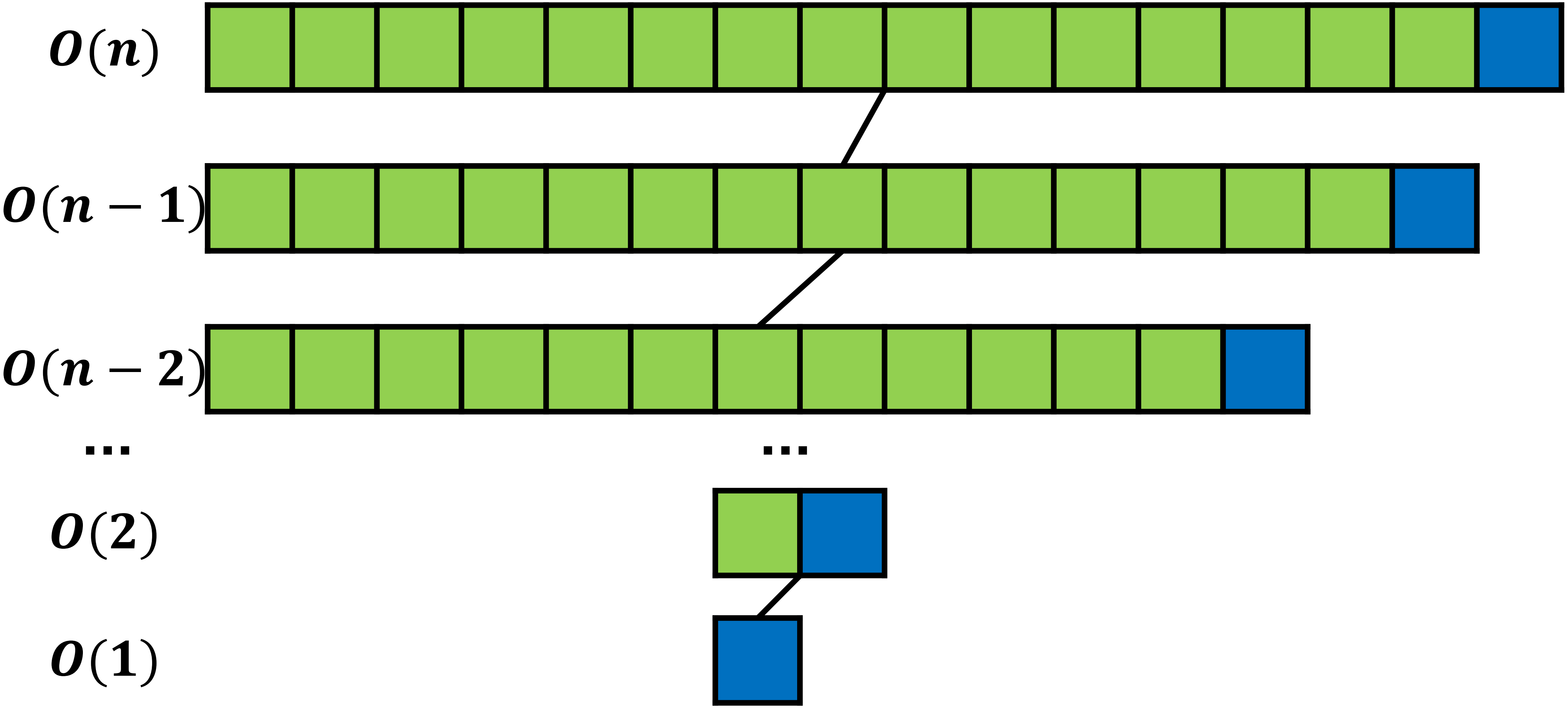
$O(n)$



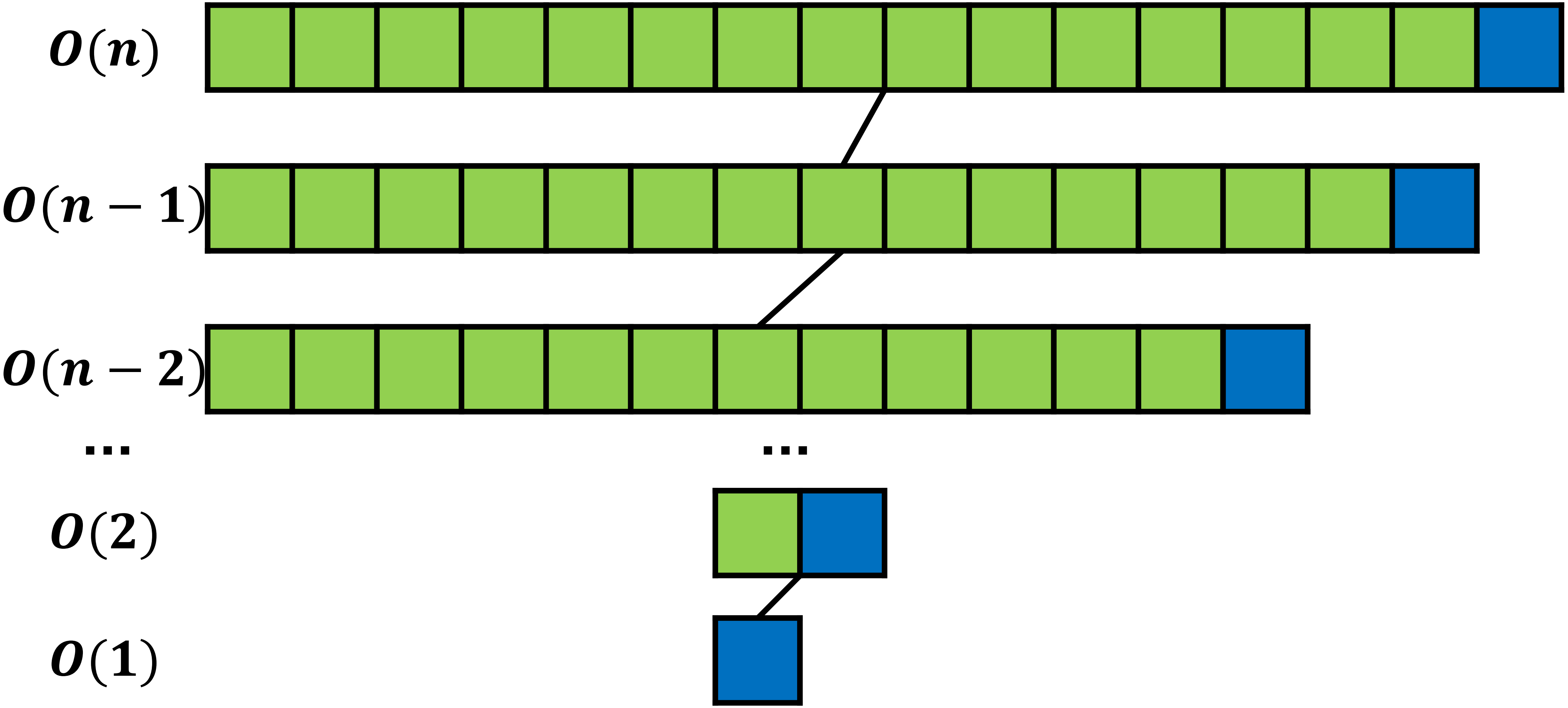
复杂度分析：最坏情况



复杂度分析：最坏情况



复杂度分析：最坏情况

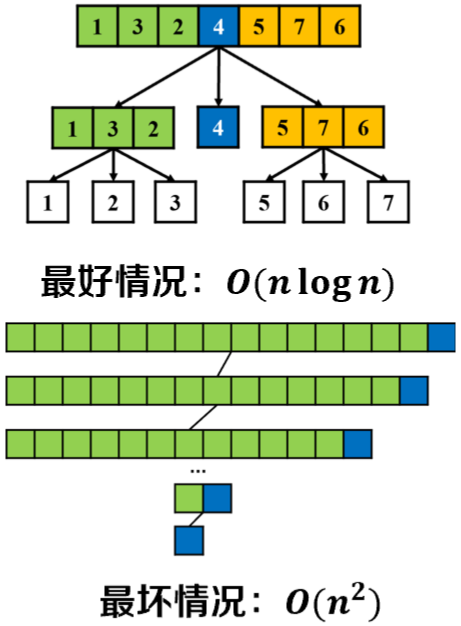


- 时间复杂度：$T(n) = \sum_{i=1}^{n} i \leq n^2 = O(n^2)$

复杂度比较

算法名称	最好情况复杂度	最坏情况复杂度
固定位置快速排序	$O(n\log n)$	$O(n^2)$
固定位置次序选择		

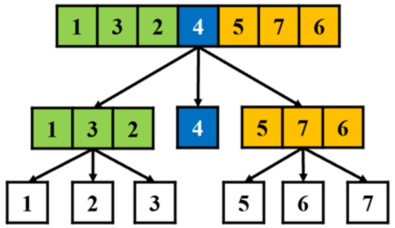
快速排序



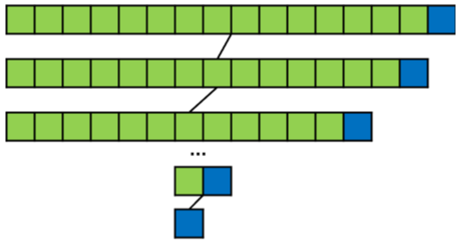
复杂度比较

算法名称	最好情况复杂度	最坏情况复杂度
固定位置快速排序	$O(n\log n)$	$O(n^2)$
固定位置次序选择	$O(n)$	$O(n^2)$

快速排序



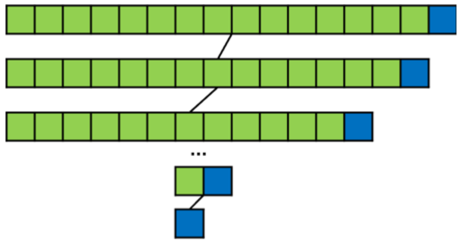
最好情况：$O(n\log n)$



最坏情况：$O(n^2)$

次序选择

最好情况：$O(n)$



最坏情况：$O(n^2)$

复杂度比较

算法名称	最好情况复杂度	最坏情况复杂度
固定位置快速排序	$O(n\log n)$	$O(n^2)$
固定位置次序选择	$O(n)$	$O(n^2)$

复杂度比较

算法名称	最好情况复杂度	最坏情况复杂度
固定位置快速排序	$O(n\log n)$	$O(n^2)$
固定位置次序选择	$O(n)$	$O(n^2)$

问题：如何摆脱最坏情况的困境？

复杂度比较

算法名称	最好情况复杂度	最坏情况复杂度
固定位置快速排序	$O(n\log n)$	$O(n^2)$
固定位置次序选择	$O(n)$	$O(n^2)$

问题：如何摆脱最坏情况的困境？

使用**随机位置**划分

随机位置划分：伪代码

- **Randomized-Partition(A, p, r)**

输入: 数组A, 起始位置p, 终止位置r
输出: 划分位置q
$s \leftarrow \mathrm{Random}(p, r)$
exchange $A[s]$ with $A[r]$
$q \leftarrow \mathrm{Partition}(A, p, r)$
return q

随机选择主元

随机位置次序选择：伪代码

- **Randomized-Selection(A, p, r, k)**

输入: 数组A,起始位置p,终止位置r,元素次序k
输出: 第k小元素x

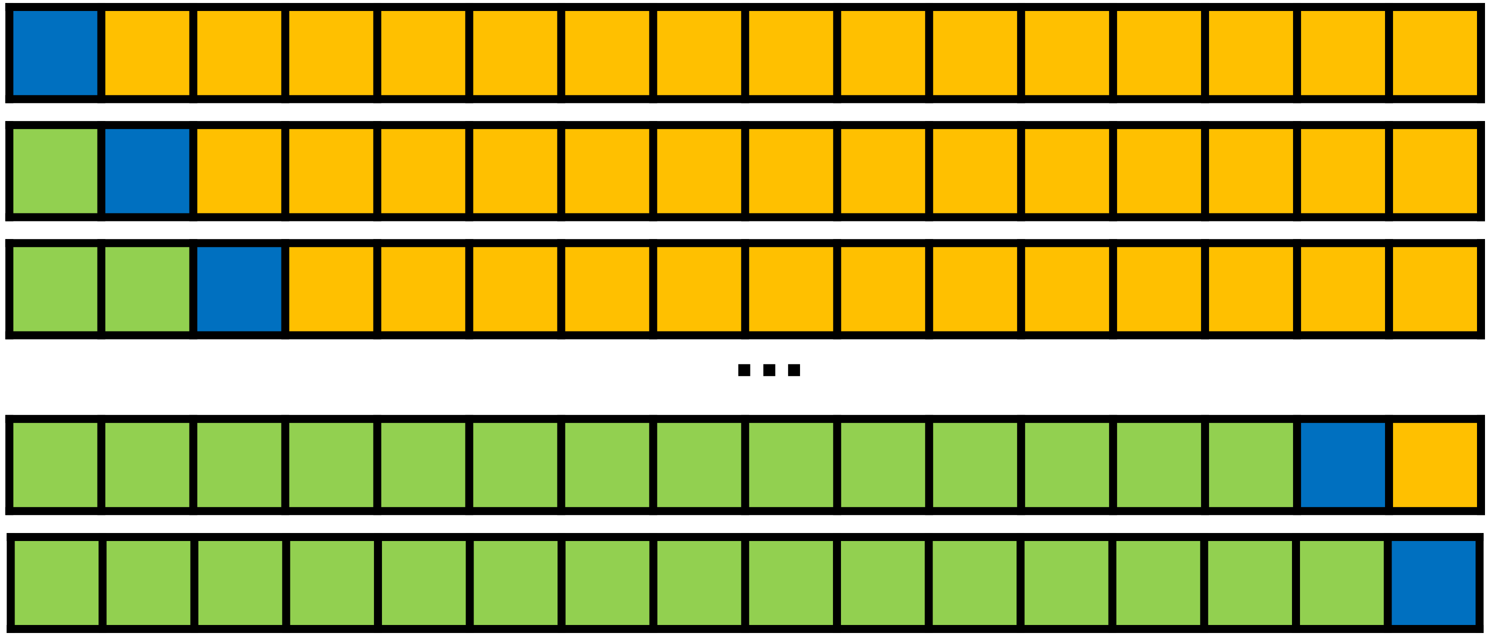
$q \leftarrow \text{Randomized-Partition}(A, p, r)$

随机划分数组

if $k = (q - p + 1)$ **then**
  $x \leftarrow A[q]$
end
if $k < (q - p + 1)$ **then**
  $x \leftarrow \text{Randomized-Selection}(A, p, q - 1, k)$
end
if $k > (q - p + 1)$ **then**
  $x \leftarrow \text{Randomized-Selection}(A, q + 1, r, k - (q - p + 1))$
end
return x

复杂度分析：期望情况

- 随机选择主元，共n种情况



0：$n-1$

1：$n-2$

2：$n-3$

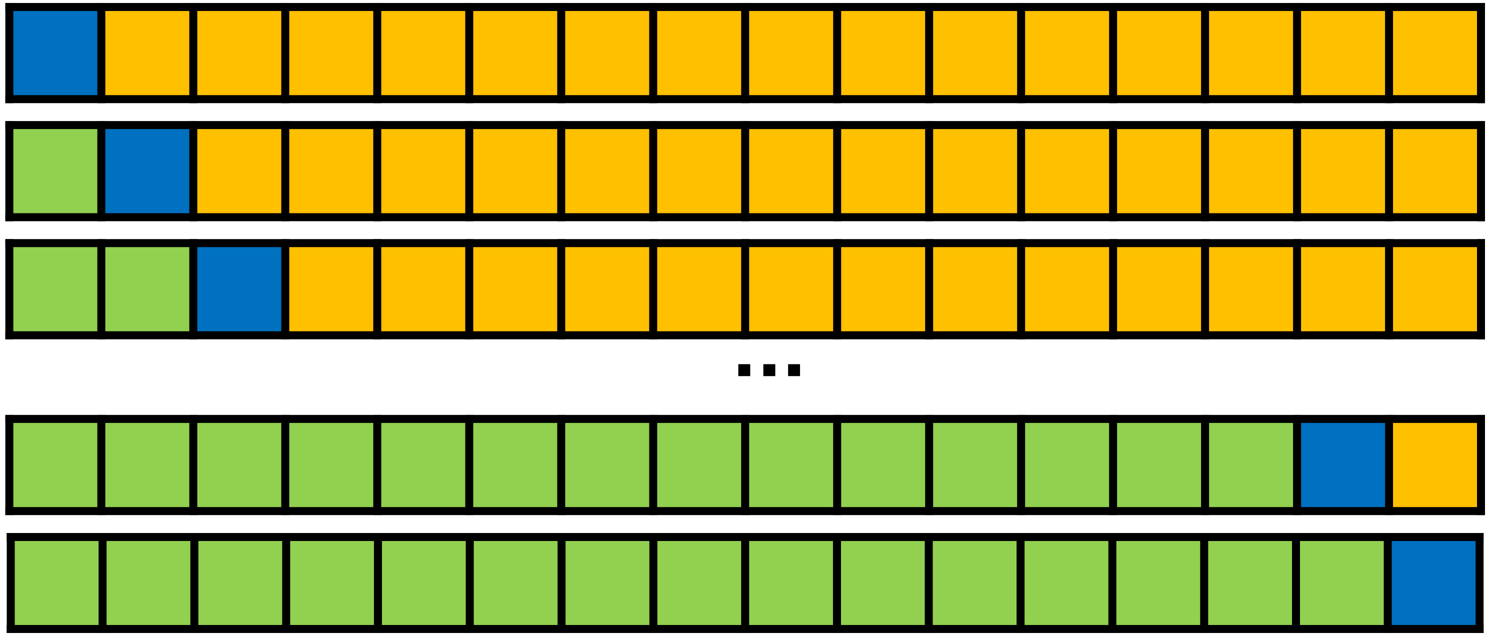
…

$n-2$：1

$n-1$：0

复杂度分析：期望情况

- 随机选择主元，共n种情况



0： $n-1$

1： $n-2$

2： $n-3$

…

$n-2$： 1

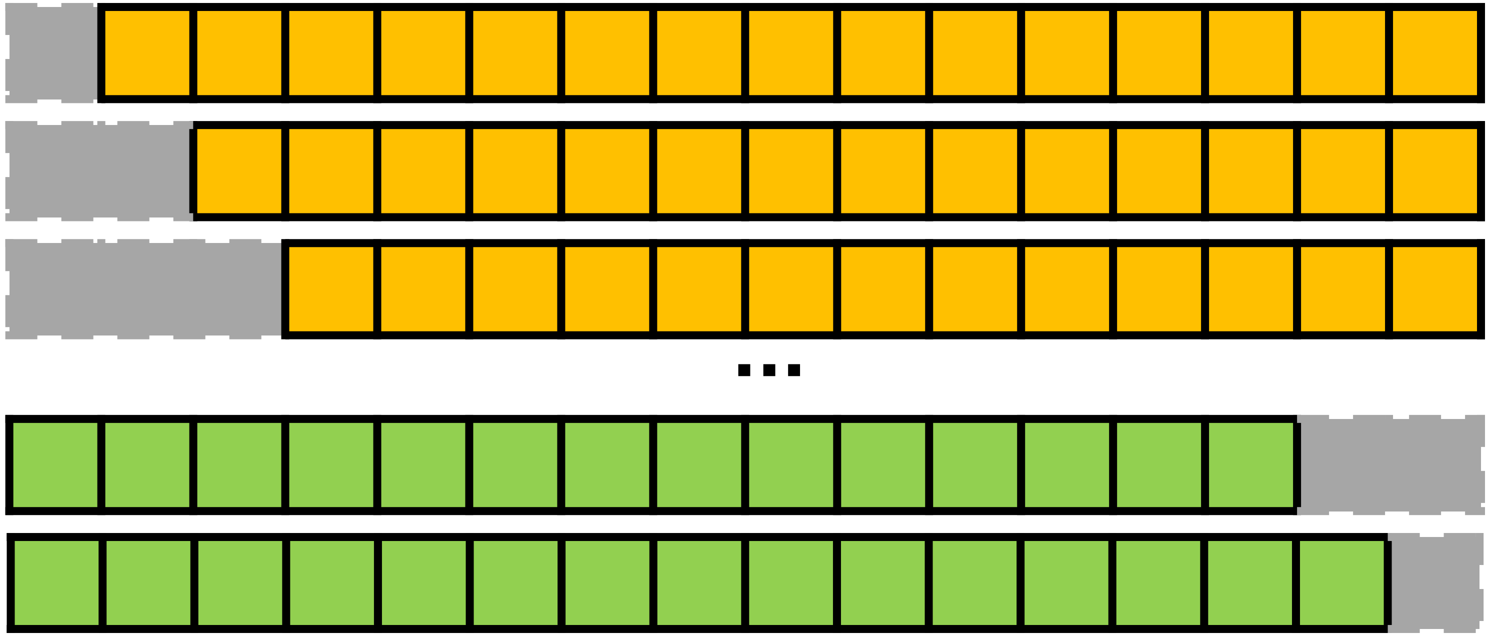
$n-1$： 0

- $$T(n) \le \begin{cases} \max\{T(0), T(n-1)\} + O(n) \\ \max\{T(1), T(n-2)\} + O(n) \\ \max\{T(2), T(n-3)\} + O(n) \\ \qquad\qquad\cdots \\ \max\{T(n-1), T(0)\} + O(n) \end{cases}$$

取较长的一段进行分析

复杂度分析：期望情况

- 随机选择主元，共n种情况



0：$n-1$

1：$n-2$

2：$n-3$

$\cdots$

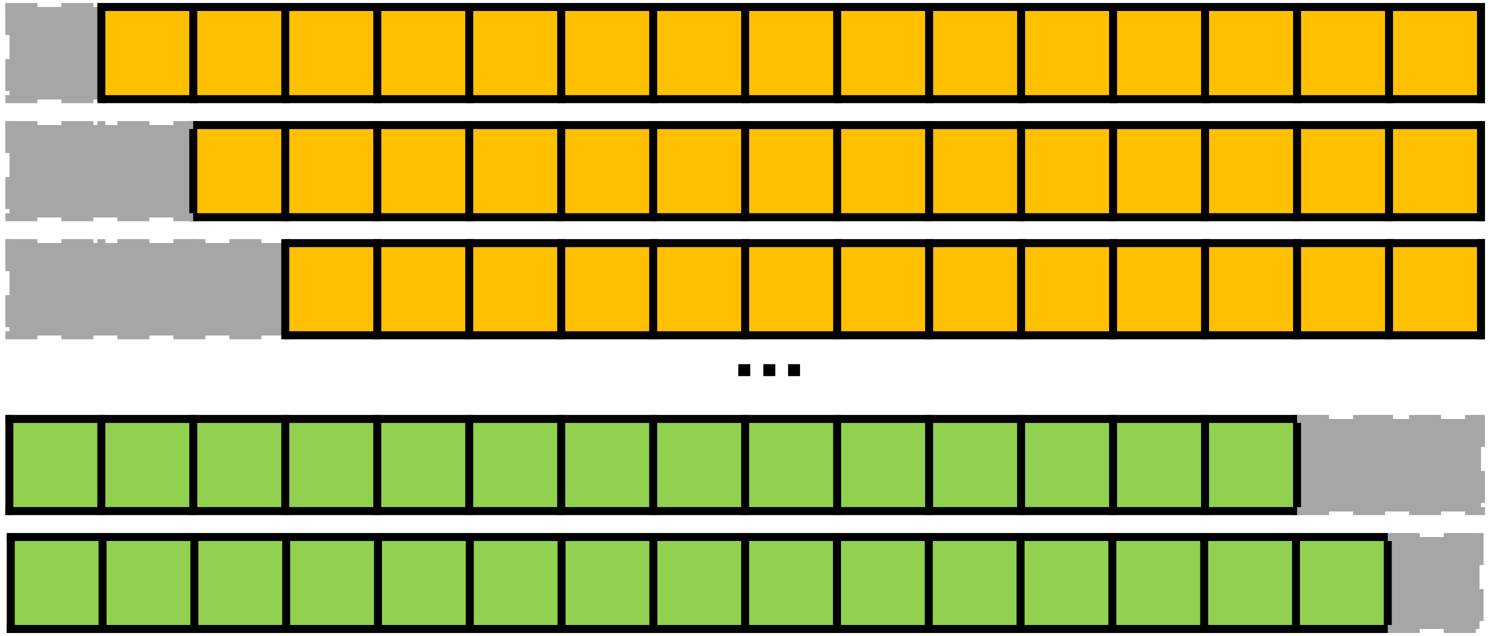
$n-2$：1

$n-1$：0

- $$T(n) \le \begin{cases} T(n-1)+O(n) \\ T(n-2)+O(n) \\ T(n-3)+O(n) \\ \qquad\cdots \\ T(n-1)+O(n) \end{cases}$$

复杂度分析：期望情况

- 随机选择主元，共n种情况



0：$n-1$

1：$n-2$

2：$n-3$

…

$n-2$：1

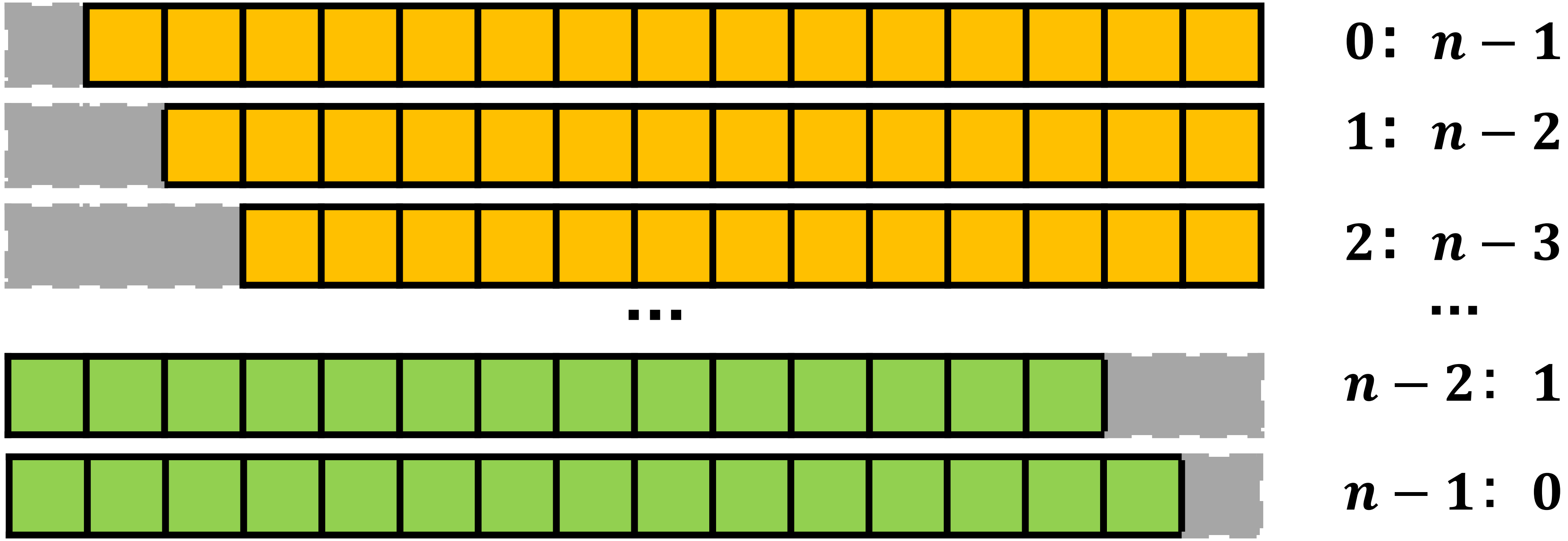
$n-1$：0

- $$T(n) \le \begin{cases} T(n-1)+O(n) \\ T(n-2)+O(n) \\ T(n-3)+O(n) \\ \quad\quad \cdots \\ T(n-1)+O(n) \end{cases}$$

n种情况概率均为$1/n$

复杂度分析：期望情况

- 随机选择主元，共n种情况



- $T(n) \le \begin{cases} T(n-1)+O(n) \\ T(n-2)+O(n) \\ T(n-3)+O(n) \\ \quad\cdots \\ T(n-1)+O(n) \end{cases}$

n种情况概率均为$1/n$
每个值$T(i)$出现2次，$i \ge \left\lfloor \frac{n}{2} \right\rfloor$

复杂度分析：期望情况

- **期望时间：**

$$E[T(n)] \le E[\frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} \left(T(i) + O(n)\right)]$$

复杂度分析：期望情况

- **期望时间：**

$$E[T(n)] \leq E[\frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} \left(T(i) + O(n)\right)]$$

每个值出现2次，概率均为$1/n$

复杂度分析：期望情况

- 期望时间：

$$E[T(n)] \leq E[\frac{2}{n} \cdot \sum_{i=\lfloor\frac{n}{2}\rfloor}^{n-1} \left(T(i) + O(n)\right)]$$

- $T(n) \leq \begin{cases} T(n-1) + O(n) \\ T(n-2) + O(n) \\ T(n-3) + O(n) \\ \dots \\ T\left(\left\lfloor\frac{n}{2}\right\rfloor\right) + O(n) \end{cases}$

复杂度分析：期望情况

- **期望时间：**

$$E[T(n)] \le E[\frac{2}{n} \cdot \sum_{i=\left\lfloor\frac{n}{2}\right\rfloor}^{n-1} \left(T(i) + O(n)\right)]$$

$$\le \frac{2}{n} \cdot \sum_{i=\left\lfloor\frac{n}{2}\right\rfloor}^{n-1} E[\left(T(i) + O(n)\right)]$$

期望的线性特性

复杂度分析：期望情况

- **期望时间：**

$$E[T(n)] \le E[\frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} \left(T(i) + O(n)\right)]$$

$$\le \frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(i)] + \frac{2}{n} \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} O(n)$$

期望的线性特性

复杂度分析：期望情况

- **期望时间：**

$$E[T(n)] \leq E[\frac{2}{n} \cdot \sum_{i=\left\lfloor \frac{n}{2} \right\rfloor}^{n-1} \left(T(i) + O(n)\right)]$$

$$\leq \frac{2}{n} \cdot \sum_{i=\left\lfloor \frac{n}{2} \right\rfloor}^{n-1} E[T(i)] + \frac{2}{n} \sum_{i=\left\lfloor \frac{n}{2} \right\rfloor}^{n-1} O(n)$$

$$\leq \frac{2}{n} \cdot \sum_{i=\left\lfloor \frac{n}{2} \right\rfloor}^{n-1} E[T(i)] + O(n)$$

$$\frac{2}{n} \sum_{i=\left\lfloor \frac{n}{2} \right\rfloor}^{n-1} O(n) = O(n)$$

复杂度分析：期望情况

- **期望时间：**

$$E[T(n)] \le E[\frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} \left(T(i) + O(n)\right)]$$

$$\le \frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(i)] + \frac{2}{n} \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} O(n)$$

$$\le \frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(i)] + O(n)$$

问题：如何进一步求解该递归式?

复杂度分析：期望情况

算法名称	最好时间复杂度	最坏时间复杂度	期望时间复杂度
快速排序	$O(n\log n)$	$O(n^2)$	$O(n\log n)$
次序选择	$O(n)$	$O(n^2)$	?

快速排序：期望时间复杂度=最好时间复杂度

复杂度分析：期望情况

算法名称	最好时间复杂度	最坏时间复杂度	期望时间复杂度
快速排序	$O(n\log n)$	$O(n^2)$	$O(n\log n)$
次序选择	$O(n)$	$O(n^2)$	$O(n)$

问题：次序选择期望时间复杂度是否为$O(n)$？

复杂度分析：期望情况

算法名称	最好时间复杂度	最坏时间复杂度	期望时间复杂度
快速排序	$O(n\log n)$	$O(n^2)$	$O(n\log n)$
次序选择	$O(n)$	$O(n^2)$	$O(n)$

问题：次序选择期望时间复杂度是否为$O(n)$?

使用代入法验证

代入法分析：期望情况

- 最好情况：$T(n) = O(n)$
- 假设：$\forall i < n,\ E[T(i)] \leq c \cdot i$

代入法分析：期望情况

- 最好情况：$T(n) = O(n)$
- 假设：$\forall i < n,\ E[T(i)] \le c \cdot i$

$$E[T(n)] \le O(n) + \frac{2}{n} \cdot \sum_{i=\left\lfloor \frac{n}{2} \right\rfloor}^{n-1} E[T(i)]$$

代入法分析：期望情况

- 最好情况：$T(n) = O(n)$
- 假设：$\forall i < n,\ E[T(i)] \le c \cdot i$

$$E[T(n)] \le O(n) + \frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(i)]$$

$$\le O(n) + \frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} c \cdot i$$

代入假设

代入法分析：期望情况

- 最好情况：$T(n) = O(n)$
- 假设：$\forall i < n,\ E[T(i)] \le c \cdot i$

$$E[T(n)] \le O(n) + \frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(i)]$$

$$\le O(n) + \frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} c \cdot i$$

$$\le O(n) + \frac{2}{n} \cdot c \cdot \frac{3}{8} n^2$$

使用不等式$\sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} i \le \frac{3}{8} n^2$

代入法分析：期望情况

- 最好情况：$T(n) = O(n)$
- 假设：$\forall i < n,\ E[T(i)] \le c \cdot i$

$$
\begin{aligned}
E[T(n)] &\le O(n) + \frac{2}{n} \cdot \sum_{i=\left\lfloor \frac{n}{2} \right\rfloor}^{n-1} E[T(i)] \\
&\le O(n) + \frac{2}{n} \cdot \sum_{i=\left\lfloor \frac{n}{2} \right\rfloor}^{n-1} c \cdot i \\
&\le O(n) + \frac{2}{n} \cdot c \cdot \frac{3}{8} n^2 \\
&= O(n) + c \cdot \frac{3}{4} n
\end{aligned}
$$

整理表达式

代入法分析：期望情况

- 最好情况：$T(n)=O(n)$
- 假设：$\forall i<n$，$E[T(i)]\le c\cdot i$

$$
\begin{aligned}
E[T(n)] &\le O(n)+\frac{2}{n}\cdot\sum_{i=\left\lfloor\frac{n}{2}\right\rfloor}^{n-1}E[T(i)]\\
&\le O(n)+\frac{2}{n}\cdot\sum_{i=\left\lfloor\frac{n}{2}\right\rfloor}^{n-1}c\cdot i\\
&\le O(n)+\frac{2}{n}\cdot c\cdot\frac{3}{8}n^2\\
&= c\cdot n-\left(\frac{1}{4}c\cdot n-O(n)\right)
\end{aligned}
$$

整理表达式

代入法分析：期望情况

- 最好情况：$T(n) = O(n)$
- 假设：$\forall i < n,\ E[T(i)] \le c \cdot i$

$$\begin{aligned}
E[T(n)] &\le O(n) + \frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(i)] \\
&\le O(n) + \frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} c \cdot i \\
&\le O(n) + \frac{2}{n} \cdot c \cdot \frac{3}{8} n^2 \\
&= c \cdot n - \left(\frac{1}{4} c \cdot n - O(n) \right) \\
&\le c \cdot n
\end{aligned}$$

选择足够大的常数c

$c \cdot \frac{n}{4}$ 渐近大于$O(n)$

代入法分析：期望情况

- 最好情况：$T(n) = O(n)$
- 假设：$\forall i < n$, $E[T(i)] \le c \cdot i$

$$
\begin{aligned}
E[T(n)] &\le O(n) + \frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(i)] \\
&\le O(n) + \frac{2}{n} \cdot \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n-1} c \cdot i \\
&\le O(n) + \frac{2}{n} \cdot c \cdot \frac{3}{8} n^2 \\
&= c \cdot n - \left(\frac{1}{4} c \cdot n - O(n) \right) \\
&\le c \cdot n
\end{aligned}
$$

随机位置次序选择：期望时间复杂度为$O(n)$

小结

快速排序

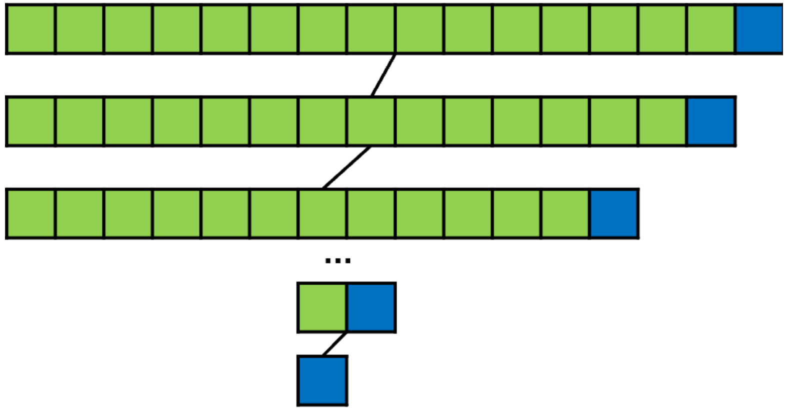
次序选择

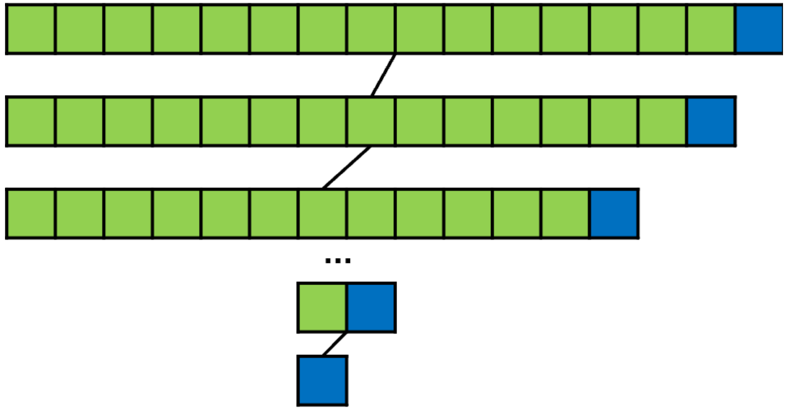
小结

快速排序

次序选择

固定位置划分





最坏情况：$O(n^2)$

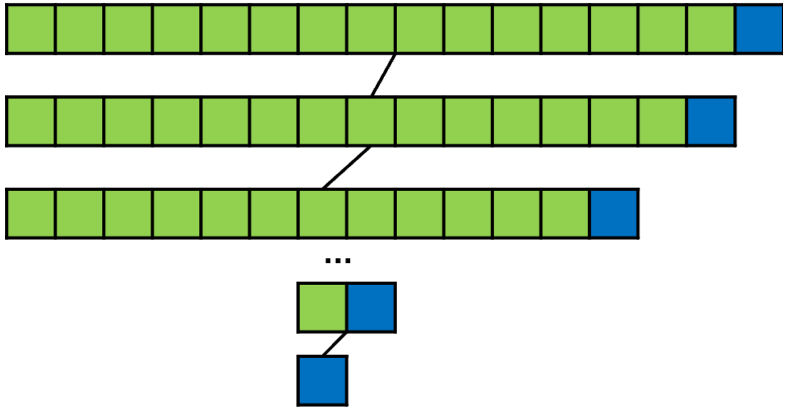
最坏情况：$O(n^2)$

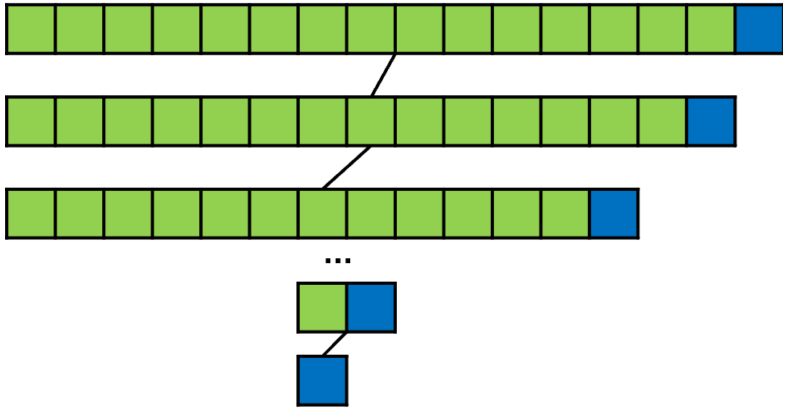
小结

快速排序

次序选择

固定位置划分



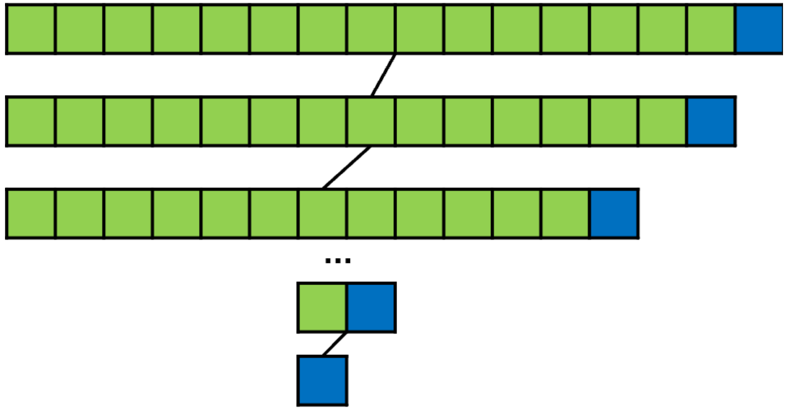
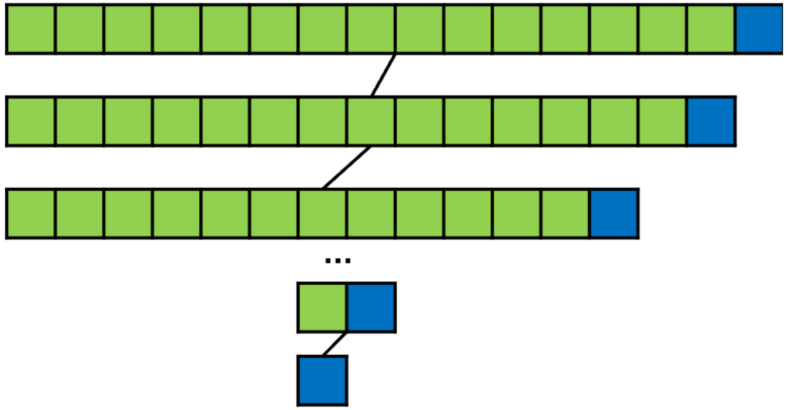
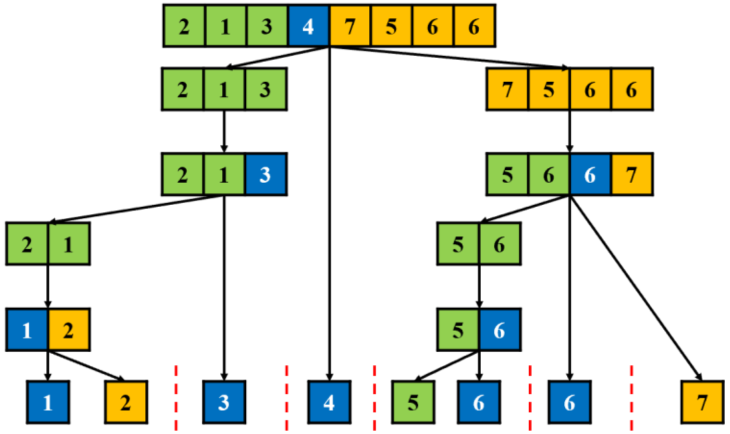
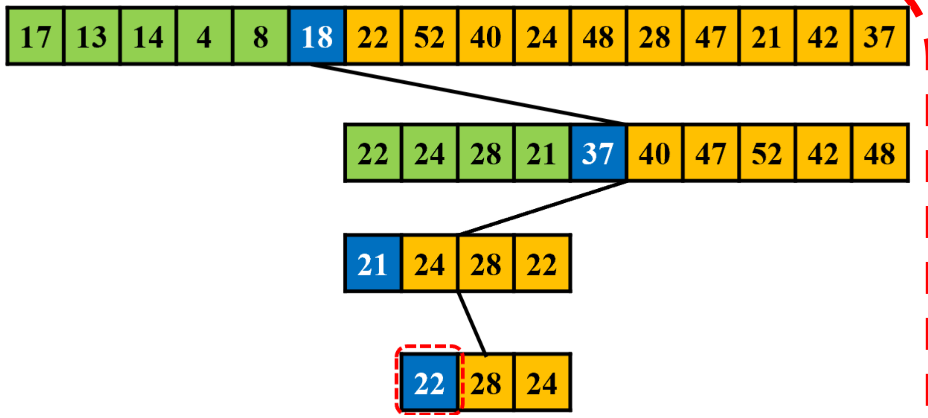


最坏情况：$O(n^2)$

最坏情况：$O(n^2)$

随机化

小结

	快速排序	次序选择
固定位置划分		
	最坏情况：$O(n^2)$	最坏情况：$O(n^2)$
随机位置划分		
	期望情况：$O(n\log n)$	期望情况：$O(n)$

总结

分而治之框架

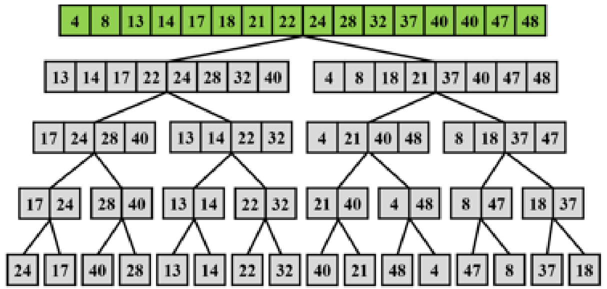
分解原问题

解决子问题

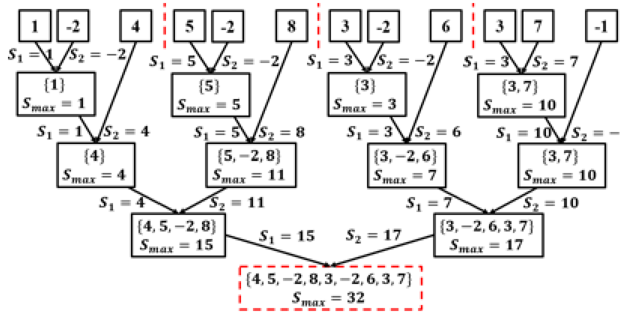
合并问题解

总结

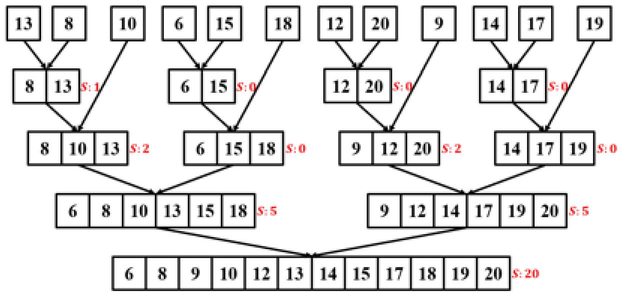
归并排序



最大子数组



逆序计数



分而治之框架

分解原问题

↓

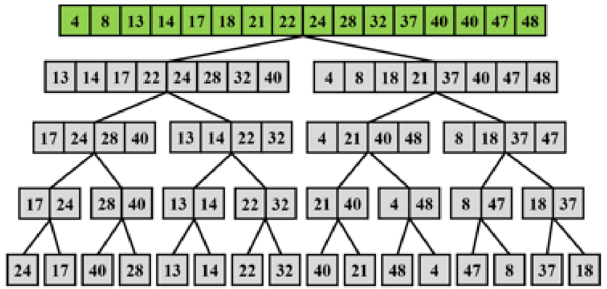
解决子问题

↓

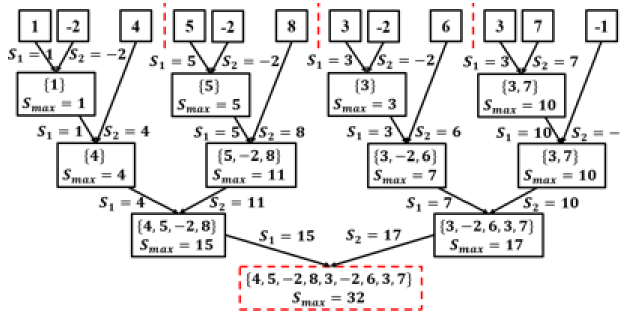
合并问题解

总结

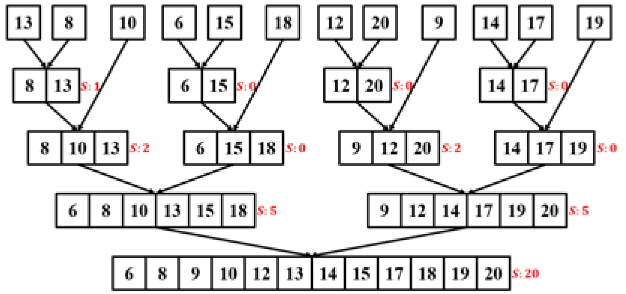
归并排序



最大子数组



逆序计数



分而治之框架

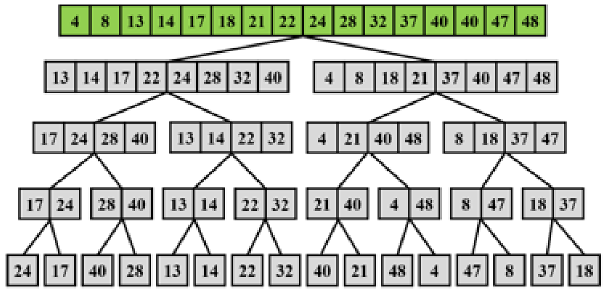
分解原问题

解决子问题

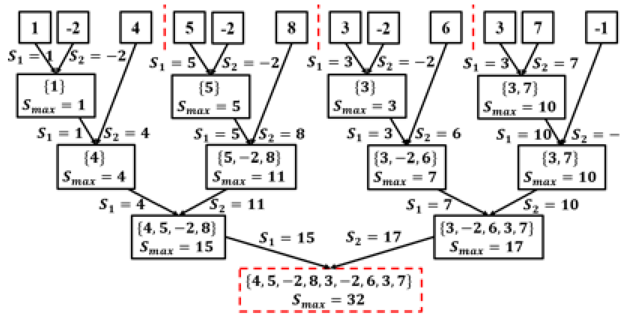
合并问题解

总结

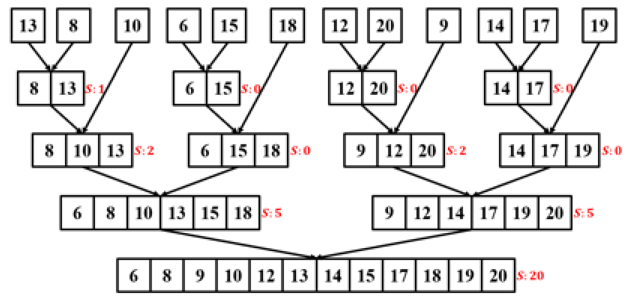
归并排序



最大子数组



逆序计数



分而治之框架

分解原问题

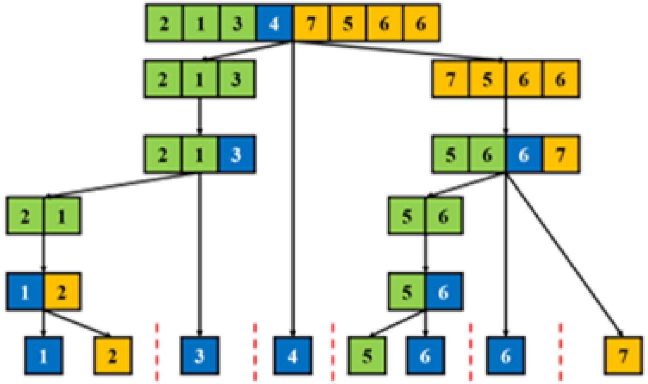
↓

解决子问题

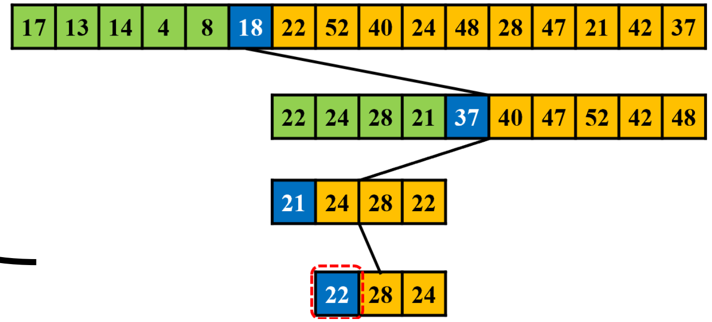
↓

合并问题解

快速排序



次序选择



谢谢

BDA

Beihang University