Selection (Order Statistics)

일을 시작하기 위해 기분이 내킬 때까지 기다리는 따위의 짓을
하지 않으려면 시험 제도는 좋은 훈련이 된다.

-아놀드 토인비

Selection: i^{th} Order Statistic

- i^{th} order statistic = i^{th} smallest element
- Want to find i^{th} order statistic out of A[p … r]
- We learn two algorithms
  - Average-case $\Theta(n)$ algorithm
  - Worst-case $\Theta(n)$ algorithm

$\Theta(n^2)$ Algorithm

Naïve approach

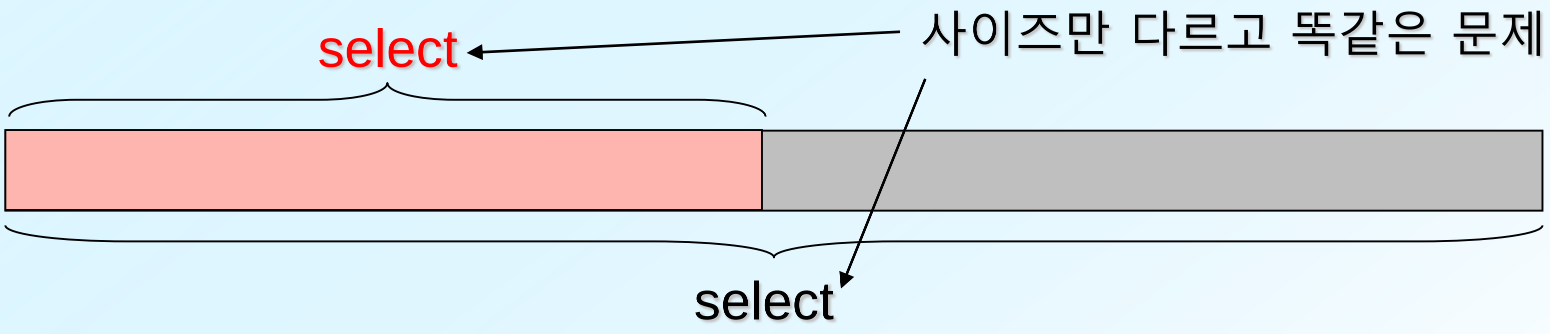
```
select (A, $p$, $r$, $i$):
◀ Find $i^{th}$ smallest in A[$p$ ... $r$]
        Find the smallest element A[$k$] in A[$p$ ... $r$]
        if ($i$ = 1)
                return A[$k$]
        else
                A[$k$ ... $r$-1] ← A[$k$+1 ... $r$]   ◀ shift left
                select (A, $p$, $r$-1, $i$-1)
```

✓Running time: $\Theta(n^2)$

Average-Case $\Theta(n)$ Algorithm

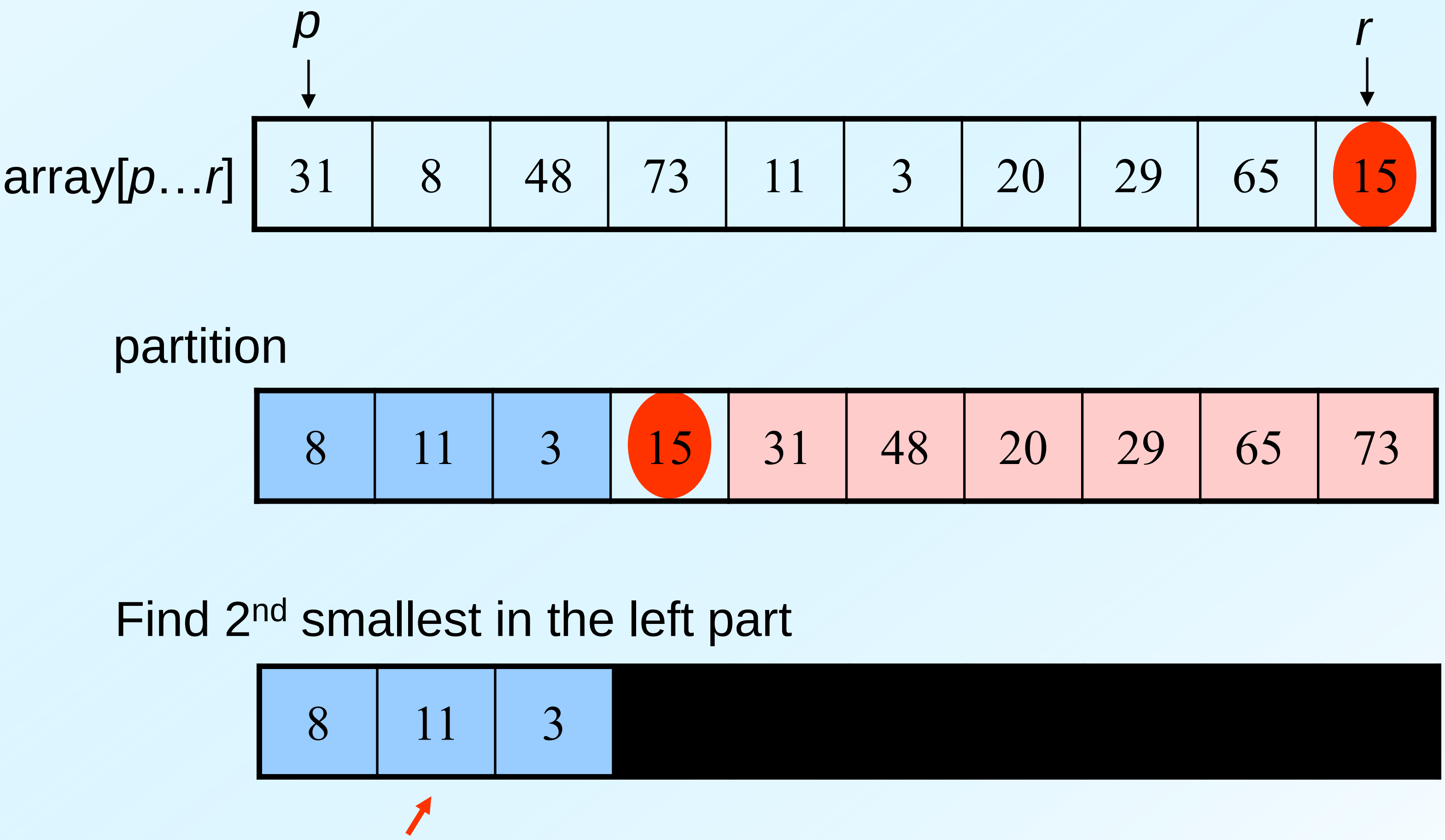
- ✓ Average-case running time: $\Theta(n)$
- ✓ Worst-case running time: $\Theta(n^2)$

```
select(A, $p$, $r$, $i$):
◀ Find $i^{th}$ smallest in A[$p$ ... $r$]
    if ($p = r$) return A[$p$]          ◀ One-element array; $i$ must be 1
    $q$ ← partition(A, $p$, $r$)
    $k$ ← $q$-$p$+1          ◀ pivot is the $k^{th}$ smallest
    if ($i < k$)
        return select(A, $p$, $q$-1, $i$)          ◀ narrow down to the left part
    else if ($i = k$)
        return A[$q$]          ◀ pivot is the $i^{th}$ smallest
    else
        return select(A, $q$+1, $r$, $i$-$k$)          ◀ narrow down to the right part
```



Example 1

Finding 2nd smallest



array[*p*…*r*]

31	8	48	73	11	3	20	29	65	15

partition

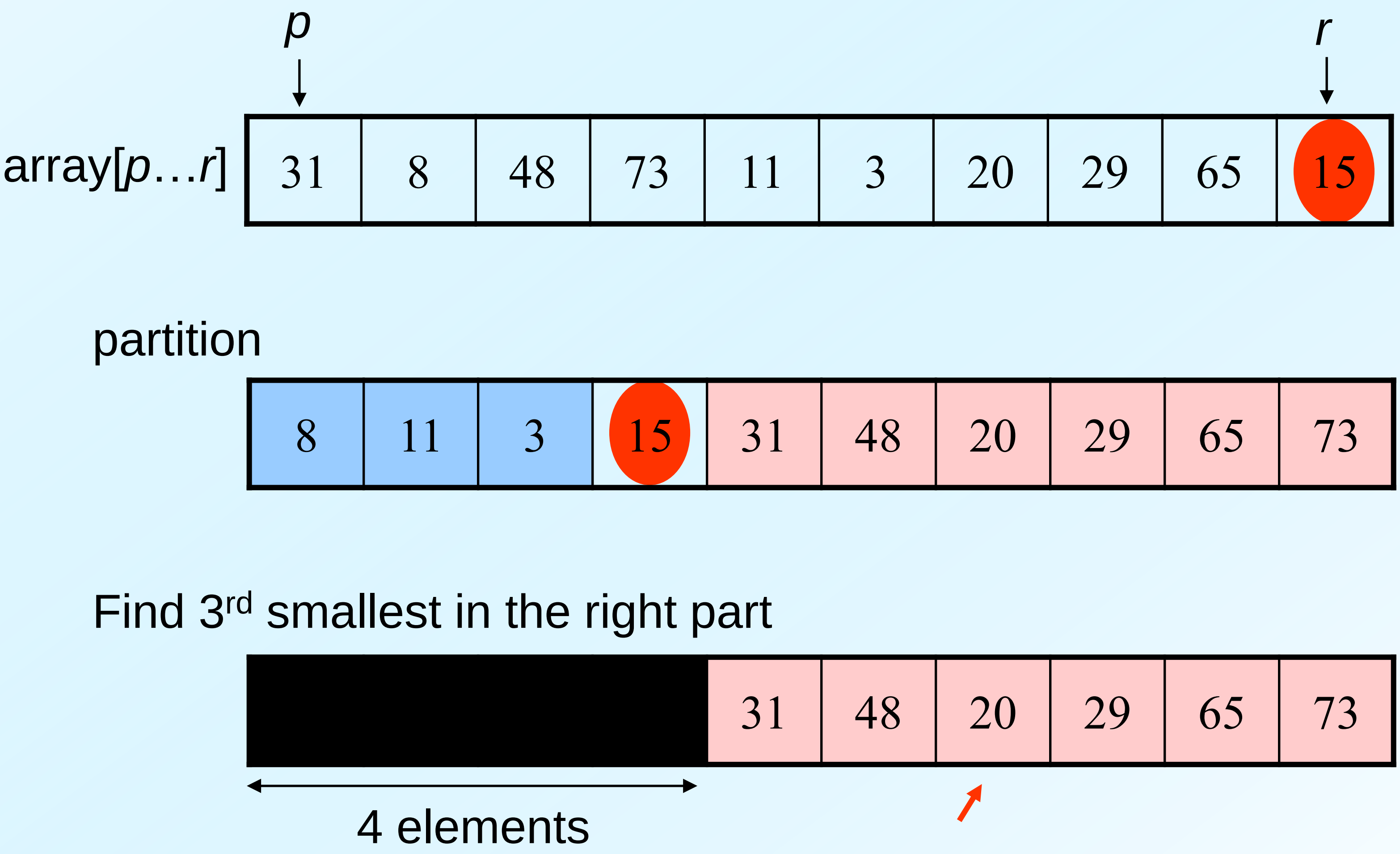
8	11	3	15	31	48	20	29	65	73

Find 2nd smallest in the left part

8	11	3	[redacted]

Example 2

Finding 7th smallest



array[*p*…*r*]

31	8	48	73	11	3	20	29	65	15

partition

8	11	3	15	31	48	20	29	65	73

Find 3rd smallest in the right part

31	48	20	29	65	73

4 elements

Average Running Time

$$T(n) \leq \frac{1}{n}\sum_{k=1}^{n} \boxed{\max[T(k\text{-}1),\ \ T(n\text{-}k)]}\ +\ \Theta(n)$$

Cost for handling the larger part
평균보다 조금 크지만 이래도 $O(n)$이 증명되면 okay

Overhead (mostly partition)

We can prove $T(n) \leq cn$ by guess&verification (next page)

$$\therefore\ T(n) = O(n)$$

Since it is obvious that $T(n) = \Omega(n),\ \ T(n) = \Theta(n)$

$$T(n) = \frac{1}{n}\sum_{k=1}^{n} T(\max(k-1, n-k)) + \Theta(n)$$

$$= \frac{2}{n}\sum_{k=\left\lfloor \frac{n}{2} \right\rfloor}^{n-1} T(k) + \Theta(n)$$

← Guess $T(n) \leq cn$
for some $c > 0, n_0 \geq 0$ and $\forall n \geq n_0$

$$\leq \frac{2}{n}\sum_{k=\left\lfloor \frac{n}{2} \right\rfloor}^{n-1} ck + \Theta(n)$$

$$= \frac{2c}{n}\left(\sum_{k=1}^{n-1} k - \sum_{k=1}^{\left\lfloor \frac{n}{2} \right\rfloor - 1} k\right) + \Theta(n)$$

$$= \frac{2c}{n}\left(\frac{(n-1)n}{2} - \frac{(\left\lfloor\frac{n}{2}\right\rfloor - 1)\left\lfloor\frac{n}{2}\right\rfloor}{2}\right) + \Theta(n)$$

$$\leq \frac{2c}{n}\left(\frac{(n-1)n}{2} - \frac{(\frac{n}{2} - 2)(\frac{n}{2} - 1)}{2}\right) + \Theta(n)$$

$$\leq c(n-1) - \frac{c}{n}\left(\frac{n^2}{4} - \frac{3n}{2} + 2\right) + \Theta(n)$$

$$\leq cn + -\frac{cn}{4} + \frac{c}{2} - \frac{2c}{n} + \Theta(n)$$

a linear function

absorbed

$$\leq cn + -\frac{cn}{4} + \Theta(n)$$

$$\leq cn$$

We can choose c>0 s.t. $-\frac{cn}{4}$ dominates $\Theta(n)$

Worst-Case Running Time

$$T(n) = T(n\text{-}1) + \Theta(n)$$

Overhead (mostly partition)

Partition $0 : n\text{-}1$, and take the larger part

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

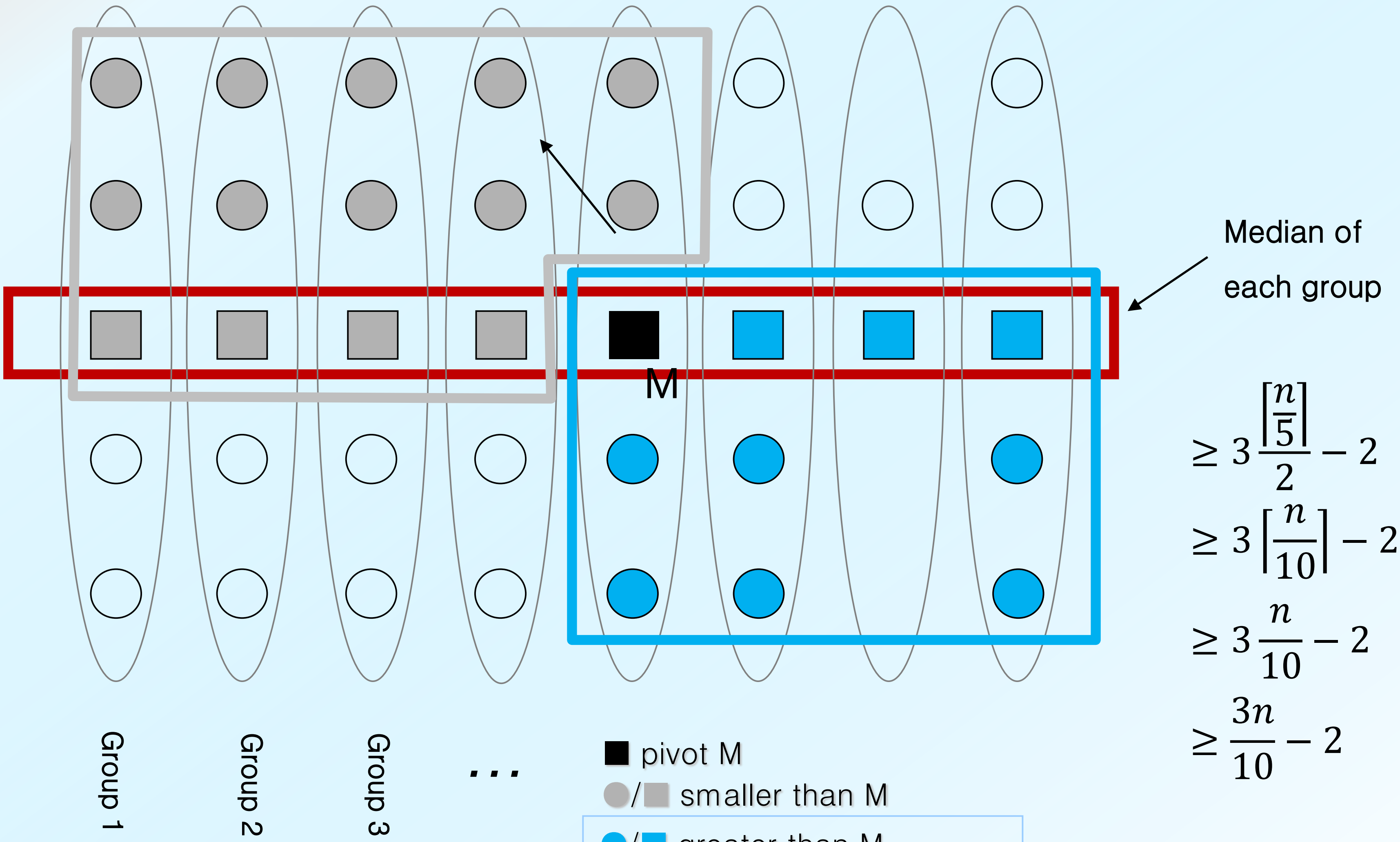
Worst-Case $\Theta(n)$ Algorithm

- We noticed from the previous algorithms that
  - the running time is affected by the balance of partition
- If always partitioned 1:1
  - $T(n) = T\left(\frac{n}{2}\right) + \Theta(n) \ \rightarrow T(n) = \Theta(n)$
- If always partitioned 3:1
  - $T(n) \leq T\left(\frac{3n}{4}\right) + \Theta(n) \ \rightarrow T(n) = O(n) \rightarrow T(n) = \Theta(n)$
- Idea
  - Restrict the worst-case partition to some extent
  - Cost for maintaining the balance should not be overly high

Worst–Case $\Theta(n)$ Algorithm

```
linearSelect(A, p, r, i):
◀ Find $i^{th}$ smallest in A[p ... r]
    ① If #elements ≤ 5, find $i^{th}$ smallest naively and finish
    ② Divide the whole set into $\lceil n/5 \rceil$ groups each having 5 elements
      (If #elements is not an exact multiple of 5, one group has fewer than 5 elements)
    ③ Find median in each group ($3^{rd}$ if #elements is 5)
       Let these medians be $m_1, m_2, \ldots, m_{\lceil n/5 \rceil}$
    ④ Find the median M of medians $m_1, m_2, \ldots, m_{\lceil n/5 \rceil}$, recursively.
       (If #elements is even, choose any of the two medians) ◀ call linearSelect( )
    ⑤ Using M as the pivot, partition the set A[p…r]
    ⑥ Choose the appropriate part and recursively repeat steps ①~⑥
                                                          ◀ call linearSelect( )
```

Relation w.r.t. the Pivot



$$\geq 3\,\frac{\left\lceil \frac{n}{5} \right\rceil}{2} - 2$$

$$\geq 3\left\lceil \frac{n}{10} \right\rceil - 2$$

$$\geq 3\,\frac{n}{10} - 2$$

$$\geq \frac{3n}{10} - 2$$

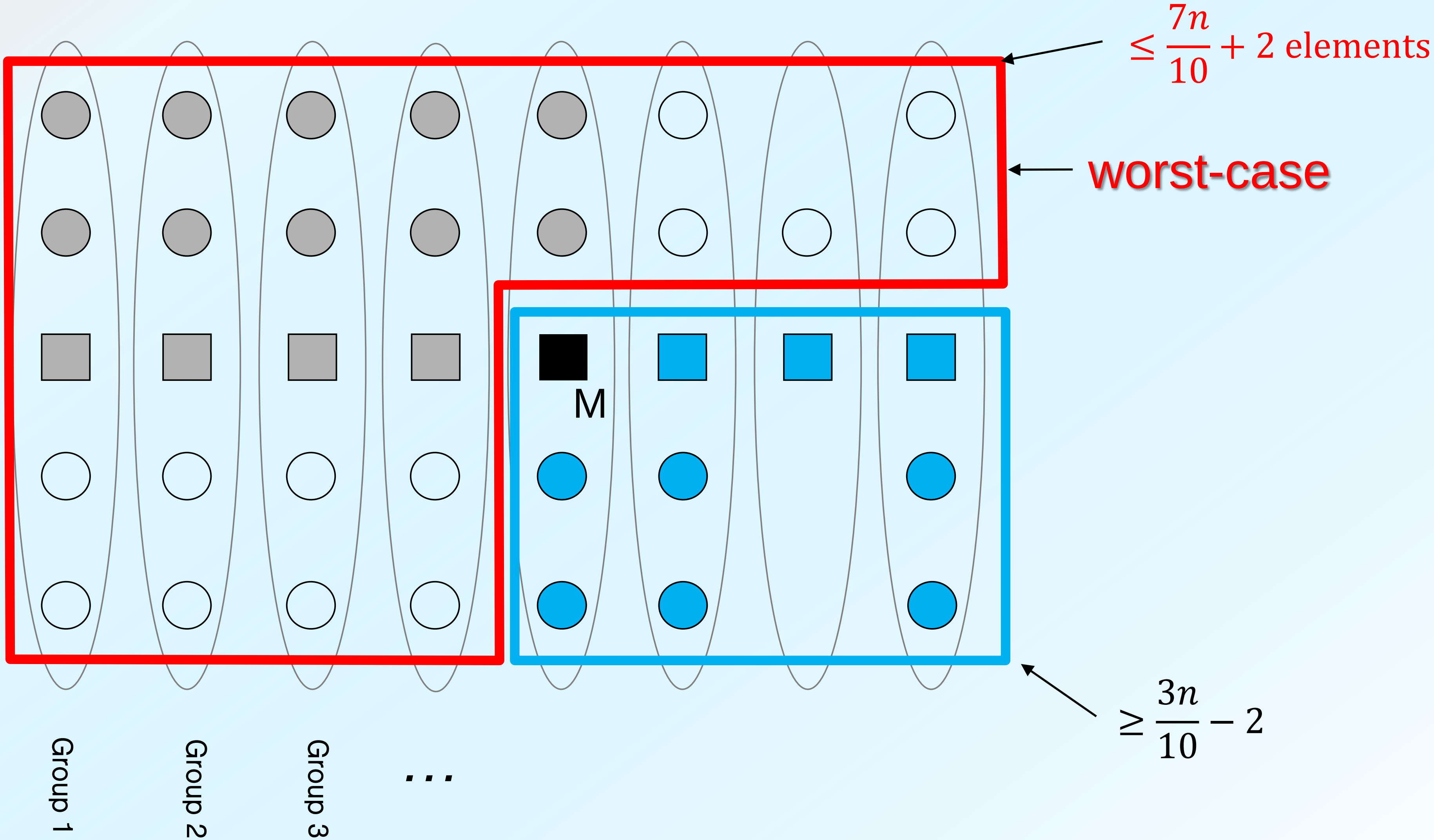
■ pivot M

●/■ smaller than M

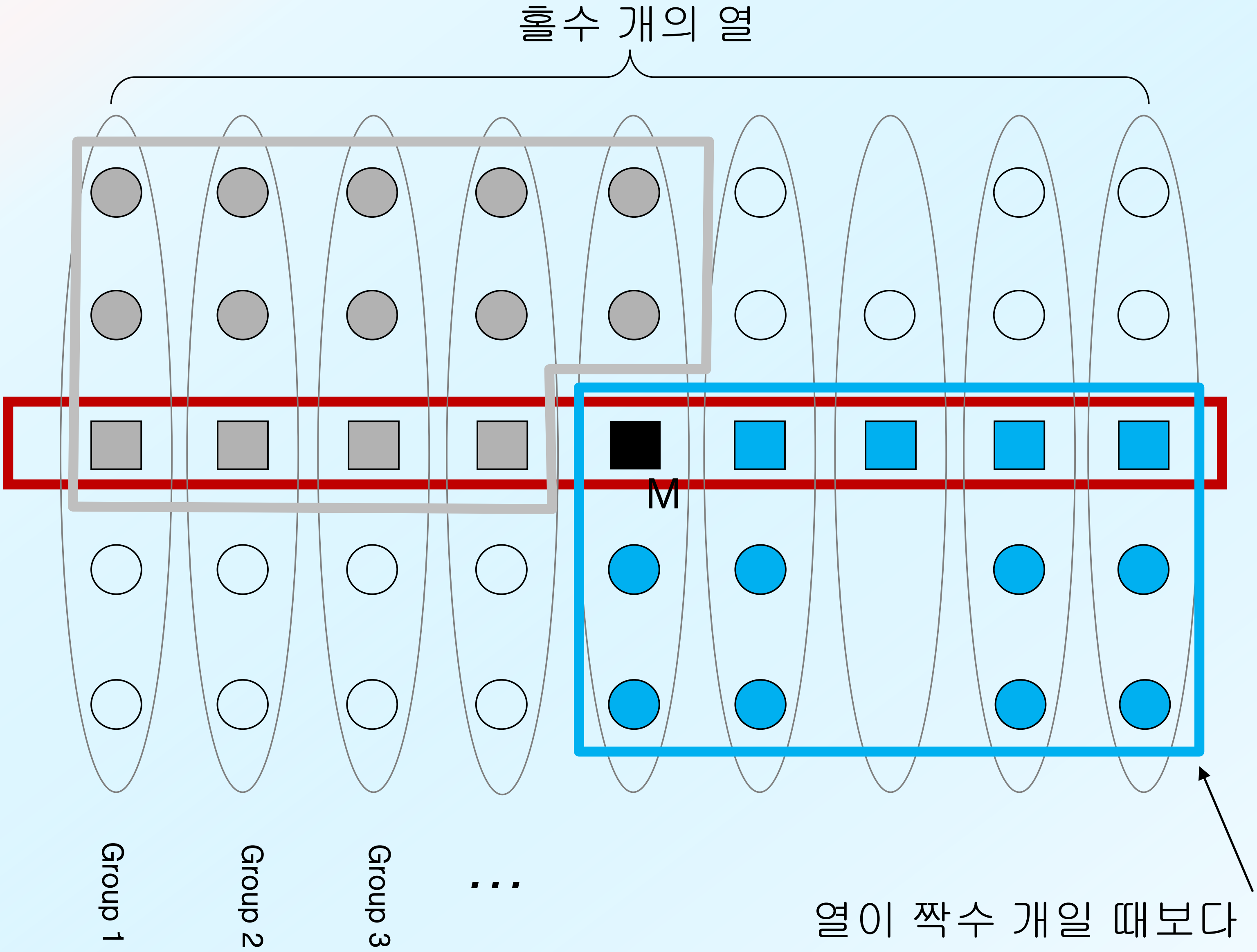
●/■ greater than M

○ may be smaller or greater than M

The Worst Case



열이 홀수 개이면



열이 짝수 개일 때보다 더 크다
(가장 나쁜 균형은 짝수일 때)

Worst-Case Running Time

Overhead for balancing

$$T(n) \le \underbrace{\boxed{T(\lceil n/5 \rceil)}}_{\text{④}} + \underbrace{T(7n/10 + 2)}_{\text{⑥}} + \underbrace{\boxed{\Theta(n)}}_{\text{①②③⑤}}$$

We can prove $T(n) \le cn$ by guess & verification (next page)

$$\therefore\ T(n) = O(n)$$

Since it is obvious that $T(n) = \Omega(n)$, $T(n) = \Theta(n)$

$$T(n) \le T\left(\left\lceil \frac{n}{5} \right\rceil\right) + T\left(\frac{7n}{10} + 2\right) + \Theta(n)$$

$$\le T\left(\frac{n}{5} + 1\right) + T\left(\frac{7n}{10} + 2\right) + \Theta(n)$$

Assume $T(k) \le ck\ \forall k, n_0 \le k < n$
(inductive assumption)

$$\le c\left(\frac{n}{5} + 1\right) + c\left(\frac{7n}{10} + 2\right) + \Theta(n)$$

$\frac{n}{5} + 1 < n$ & $\frac{7n}{10} + 2 < n$

$\rightarrow 7 \le n$

$$= c\left(\frac{9n}{10} + 3\right) + \Theta(n)$$

$$= cn - \frac{cn}{10} + 3c + \Theta(n)$$

$$\le cn$$

We can choose c>0 s.t. $-\frac{cn}{10}$ dominates $3c + \Theta(n)$