Chap 7 – Quicksort

7.1 Description of quicksort

- Quicksort

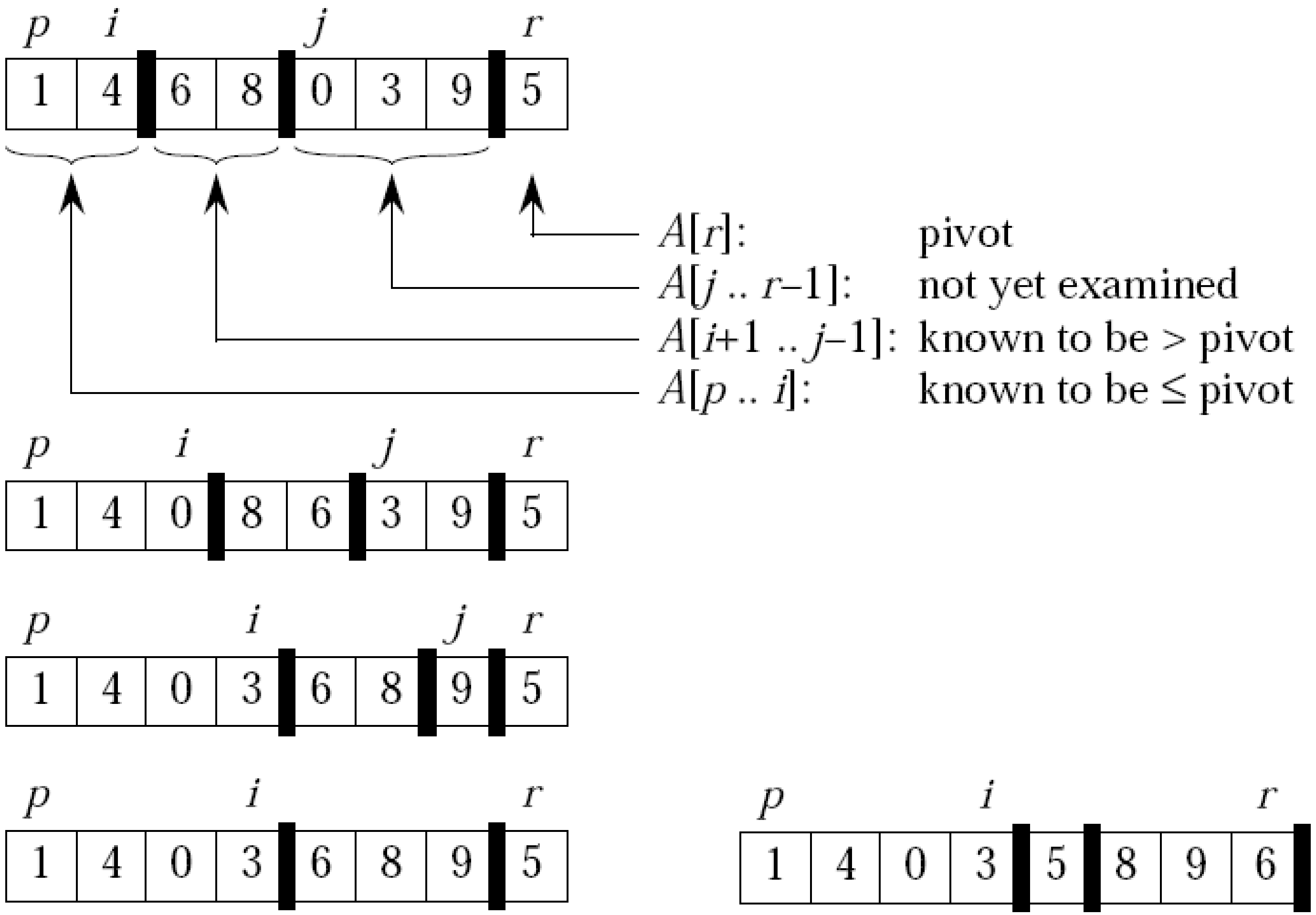
QUICKSORT(A, p, r)
if $p < r$
  $q = $ PARTITION(A, p, r)
  QUICKSORT$(A, p, q - 1)$
  QUICKSORT$(A, q + 1, r)$

PARTITION(A, p, r)
$x = A[r]$
$i = p - 1$
for $j = p$ **to** $r - 1$
  if $A[j] \leq x$
    $i = i + 1$
    exchange $A[i], A[j]$
exchange $A[i + 1], A[r]$
return $i + 1$

$\Theta(n)$ running time,
where $n = r - p + 1$

7.1 Description of quicksort

- Partition – an example



7.2 Performance of quicksort

- Worst-case partitioning
  - Maximally unbalanced
  - Worst-case input: when the elements are already sorted.
  - $T(n) = T(0) + T(n-1) + \Theta(n)$
    $\Rightarrow T(n) = T(n-1) + \Theta(n) \Rightarrow T(n) = \Theta(n^2)$
- Best-case partitioning
  - Equally balanced
  - $T(n) \leq 2T(n/2) + \Theta(n) \Rightarrow T(n) = O(n \lg n)$
    since the two subarrays have $n - 1 < n$ elements in total.
    or,
    $T(n) = 2T((n-1)/2) + \Theta(n) \Rightarrow T(n) = \Theta(n \lg n)$

7.2 Performance of quicksort

- Balanced partitioning

  Imagine that PARTITION always produces a 1-to-9 split

  Get the recurrence

  $T(n) \leq T(n/10) + T(9n/10) + \Theta(n) \Rightarrow T(n) = O(n \lg n)$

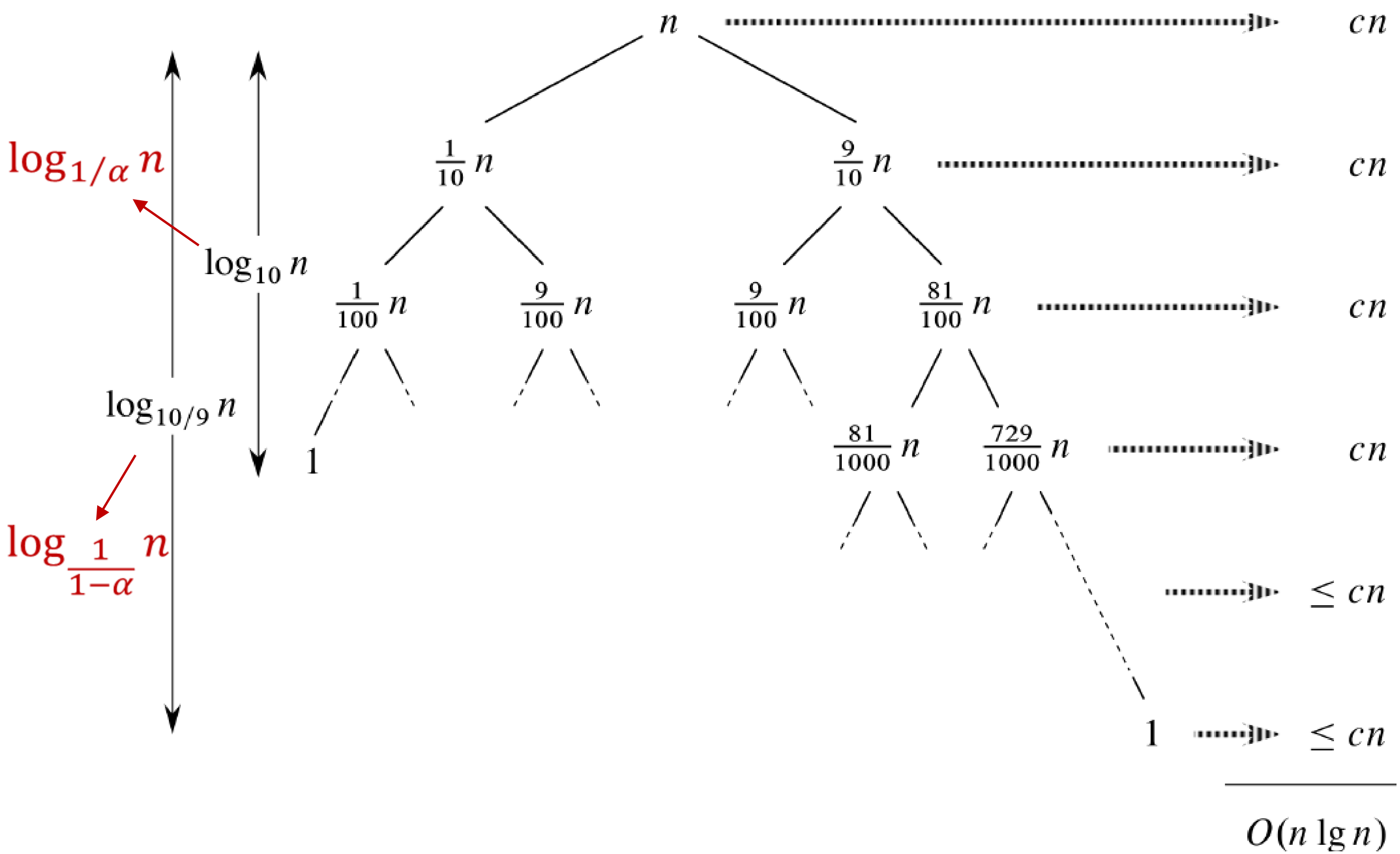
  In general, if PARTITION always produces an $\alpha : 1 - \alpha$ split, where $0 < \alpha \leq 1/2$, we have

  $T(n) \leq T(\alpha n) + T((1-\alpha)n) + \Theta(n) \Rightarrow T(n) = O(n \lg n)$

  It is reasonable to guess that the average case is closer to the best case.

7.2 Performance of quicksort

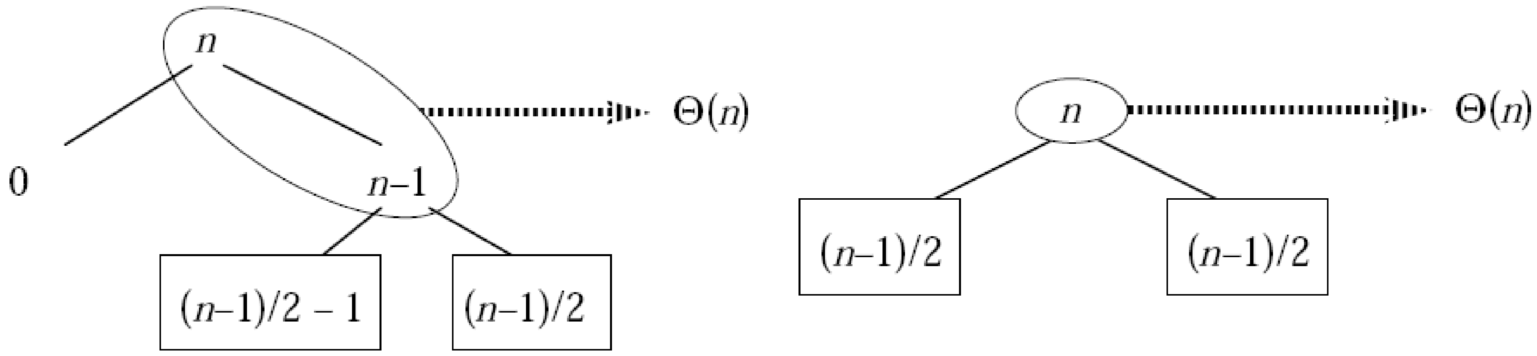
- Balanced partitioning



7.2 Performance of quicksort

- Intuition for the average case
  - Usually a mix of good and bad splits
  - Don't affect the asymptotic running time

Good and bad splits alternate   Good split alone



Both spent $\Theta(n)$ partitioning time

Both have nearly the same subproblems to solve

7.3 A randomized version of quicksort

- Random sampling

  Choose the pivot unbiasedly at random from $A[p..r]$

```
RANDOMIZED-PARTITION(A, p, r)
i = RANDOM(p, r)
exchange A[r] with A[i]
return PARTITION(A, p, r)
```

```
RANDOMIZED-QUICKSORT(A, p, r)
if p < r
    q = RANDOMIZED-PARTITION(A, p, r)
    RANDOMIZED-QUICKSORT(A, p, q - 1)
    RANDOMIZED-QUICKSORT(A, q + 1, r)
```

7.4. Analysis of quicksort

- Best-case analysis

  $$T(n) = \min_{0 \le k \le n-1} (T(k) + T(n-k-1)) + \Theta(n)$$

  Then, $T(n) = \Omega(n \lg n)$ See Ex.7.4-2

  Also, $T(n) = O(n \lg n)$

  The upper bound follows immediately from the fact that Quicksort takes a time in $\Theta(n \lg n)$ when the input causes balanced partitioning, because

  $T(n) =$ the minimum running time of all cases

  $\le$ the running time of any particular case

7.4. Analysis of quicksort

- Worst-case analysis

  $$T(n) = \max_{0 \le k \le n-1} (T(k) + T(n-k-1)) + \Theta(n)$$

  How to solve recurrences with max or min?

  - Guess where the max or min occurs, pay attention to $0, 1, \lfloor n/2 \rfloor, \lceil n/2 \rceil, n-1, n$
  - Solve the recurrence using the presumed max or min
  - Use the solution to prove that the guess is correct

  For the preceding recurrence

  - Guess the max occurs when $k = 0$ or $n - 1$
  - Solve $T(n) = T(0) + T(n-1) + \Theta(n) \Rightarrow T(n) = \Theta(n^2)$
  - Prove that $T(n) = \Theta(n^2)$

7.4. Analysis of quicksort

- Worst-case analysis

  Upper bound

  Assume that $T(n) \leq cn^2$

  $$T(n) = \max_{0 \leq k \leq n-1} (T(k) + T(n-k-1)) + \Theta(n)$$
  $$\leq \max_{0 \leq k \leq n-1} (ck^2 + c(n-k-1)^2) + \Theta(n)$$
  $$= c \cdot \max_{0 \leq k \leq n-1} (k^2 + (n-k-1)^2) + \Theta(n)$$

  Let $f(k) = k^2 + (n-k-1)^2$

  $f''(k) = 4 > 0$

  $\Rightarrow$ the maximum occurs at $k = 0$ or $n - 1$

7.4. Analysis of quicksort

- Worst-case analysis

  Thus,

  $$T(n) \le c(n-1)^2 + \Theta(n)$$
  $$= cn^2 - c(2n-1) + \Theta(n)$$
  $$\le cn^2$$

  $\because$ We can pick c large enough so that $c(2n-1)$ dominates $\Theta(n)$

  Lower bound

  Assume that $T(n) \ge cn^2$ and work out

  $$T(n) \ge cn^2 - c(2n-1) + \Theta(n)$$
  $$\ge cn^2$$

  $\because$ We can pick c small enough so that $\Theta(n)$ dominates $c(2n-1)$

7.4. Analysis of quicksort

- Worst-case analysis

  An alternative argument for lower bound

  Since Quicksort takes a time in $\Theta(n^2)$ when the input is already sorted, it follows that $T(n) = \Omega(n^2)$, because

  $T(n) =$ the maximum running time of all cases

  $\geq$ the running time of any particular case

- Expected running time of RANDOMIZED-QUICKSORT

  Lower bound

  Since

  average-case complexity $\geq$ best-case complexity = $\Theta(n \lg n)$

  it follows that the expected running time $T(n) = \Omega(n \lg n)$

7.4. Analysis of quicksort

- Expected running time of RANDOMIZED-QUICKSORT

  Upper bound

  - Key observation: PARTITION removes the pivot from future computation each time
  - PARTITION is called at most $k < n$ times (in the worst case $k = n - 1$)

    Each call spends $\Theta(\text{number of comparisons})$ time.
  - QUICKSORT is called at most $2k + 1$ times

    Each call spends $\Theta(1)$ time, excluding partitioning time.

7.4. Analysis of quicksort

- Expected running time of RANDOMIZED-QUICKSORT

  Let $X =$ the total number of comparisons performed in all calls to PARTITION

  LEMMA 7.1

  The running time of QUICKSORT is $O(n + X)$.

  The expected running time is $O(n + \mathrm{E}[X])$.

  Proof

  Let the elements be $z_1 < z_2 < \cdots < z_n$

  Define the set $Z_{ij} = \{z_i, z_{i+1}, \ldots, z_j\}$

  Define the indicator random variable

  $X_{ij} = \mathrm{I}\{z_i \text{ is compared to } z_j\}$

7.4. Analysis of quicksort

- Expected running time of RANDOMIZED-QUICKSORT

  Since the pivot is removed from future consideration

  $\Rightarrow$ each pair of elements is compared at most once

  $$\Rightarrow X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}$$

  Thus,

  $$\begin{aligned} \mathrm{E}[X] &= \mathrm{E}\left[\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}\right] \\ &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathrm{E}\left[X_{ij}\right] \\ &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \Pr\left\{z_i \text{ is compared to } z_j\right\} \end{aligned}$$

7.4. Analysis of quicksort

- Expected running time of RANDOMIZED-QUICKSORT

  Once a pivot x is chosen such that $z_i < x < z_j$, then z_i and z_j will never be compared at any later time.

  Therefore,

  $$\begin{aligned}
  &\Pr\{z_i \text{ is compared to } z_j\} \\
  &= \Pr\{z_i \text{or } z_j \text{ is the first pivot chosen from } Z_{ij}\} \\
  &= \Pr\{z_i \text{ is the first pivot chosen from } Z_{ij}\} \\
  &\qquad + \Pr\{z_j \text{ is the first pivot chosen from } Z_{ij}\} \\
  &= \frac{1}{j-i+1} + \frac{1}{j-i+1} \\
  &= \frac{2}{j-i+1}
  \end{aligned}$$

  Given 1, 2, 3, 4, 5

  $$\Pr\{2 \text{ is compared to } 4\} = \frac{2}{5} + 2 \times \frac{1}{5} \times \left[\frac{2}{4} + \frac{1}{4} \times \frac{2}{3}\right] = \frac{2}{3}$$

7.4. Analysis of quicksort

- Expected running time of RANDOMIZED-QUICKSORT

  Finally,

$$\mathrm{E}[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \Pr\{z_i \text{ is compared to } z_j\}$$

$$= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1}$$

$$= \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1} \quad \because k = j - i$$

$$< \sum_{i=1}^{n-1} \sum_{k=1}^{n} \frac{2}{k}$$

$$= \sum_{i=1}^{n-1} O(\lg n) = O\left(\sum_{i=1}^{n-1} \lg n\right) = O(n \lg n)$$

7.4. Analysis of quicksort

- Analyze expected running time by recurrence

  Define the indicator random variables (Problem 7-2)

  $X_k = \mathrm{I}\{\text{the pivot is the } k\text{th smallest element}\}$

  Then,

  $\mathrm{E}[X_k] = \Pr\{\text{the pivot is the } k\text{th smallest element}\} = 1/n$

  Let $T(n)$ be a random variable denoting the running time of Quicksort on n elements

  Then,

  $$T(n) = \sum_{k=1}^{n} X_k \cdot (T(k-1) + T(n-k) + \Theta(n))$$

  Note that only one of the X_k's has the value 1.

7.4. Analysis of quicksort

- Analyze expected running time by recurrence

Therefore,

$$\mathrm{E}[T(n)] = \mathrm{E}\left[\sum_{k=1}^{n} X_k \cdot (T(k-1) + T(n-k) + \Theta(n))\right]$$

$$= \sum_{k=1}^{n} \mathrm{E}[X_k \cdot (T(k-1) + T(n-k) + \Theta(n))]$$

$$= \sum_{k=1}^{n} \mathrm{E}[X_k] \cdot \mathrm{E}[T(k-1) + T(n-k) + \Theta(n)]$$

$\because$ independent

$$= \frac{1}{n} \sum_{k=1}^{n} \mathrm{E}[T(k-1) + T(n-k) + \Theta(n)]$$

7.4. Analysis of quicksort

- Analyze expected running time by recurrence

$$\mathrm{E}[T(n)] = \frac{1}{n}\sum_{k=1}^{n}(\mathrm{E}[T(k-1)] + \mathrm{E}[T(n-k)] + \Theta(n))$$

$\because$ linearity of expectation

$$= \frac{2}{n}\sum_{k=0}^{n-1}\mathrm{E}[T(k)] + \Theta(n)$$

How to solve this *full-history* recurrence?

Method A (Problem 7-3)

Prove by substitution that $\mathrm{E}[T(n)] = \Theta(n \lg n)$

Method B

Elimination of history

7.4. Analysis of quicksort

- Analyze expected running time by recurrence

$$\mathrm{E}[T(n)] = \frac{2}{n}\sum_{k=0}^{n-1}\mathrm{E}[T(k)] + \Theta(n) = \frac{2}{n}\sum_{k=0}^{n-1}\mathrm{E}[T(k)] + cn$$

Replace $\Theta(n)$ by cn

Multiply both sides by n

$$n\mathrm{E}[T(n)] = 2\sum_{k=0}^{n-1}\mathrm{E}[T(k)] + cn^2 \qquad (1)$$

Substitute $n-1$ for n in (1)

$$(n-1)\mathrm{E}[T(n-1)] = 2\sum_{k=0}^{n-2}\mathrm{E}[T(k)] + c(n-1)^2 \qquad (2)$$

7.4. Analysis of quicksort

- Analyze expected running time by recurrence

  $(1)-(2)$

  $$nE[T(n)]-(n-1)E[T(n-1)]=2E[T(n-1)]+c(2n-1)$$

  Thus,

  $$\begin{aligned}E[T(n)] &= \frac{n+1}{n}E[T(n-1)]+c\left(2-\frac{1}{n}\right)\\ &= \frac{n+1}{n}E[T(n-1)]+\Theta(1)\end{aligned}$$

  Now that the history is eliminated, it can be easily solved by repetitive substitution or range transformation.

7.4. Analysis of quicksort

- Analyze expected running time by recurrence

  Repetitive substitution

$$
\begin{aligned}
\mathrm{E}[T(n)] &= \frac{n+1}{n}\mathrm{E}[T(n-1)] + \Theta(1) \\
&= \frac{n+1}{n}\left(\frac{n}{n-1}\mathrm{E}[T(n-2)] + \Theta(1)\right) + \Theta(1) \\
&= \frac{n+1}{n-1}\mathrm{E}[T(n-2)] + \left(\frac{n+1}{n} + \frac{n+1}{n+1}\right)\Theta(1) \\
&= \cdots \\
&= \frac{n+1}{2}\mathrm{E}[T(1)] + \left(\frac{n+1}{3} + \cdots + \frac{n+1}{n} + \frac{n+1}{n+1}\right)\Theta(1) \\
&= (n+1)\left(\sum_{k=2}^{n+1}\frac{1}{k}\right)\Theta(1) = (n+1)\Theta(\lg n)\Theta(1) = \Theta(n\lg n)
\end{aligned}
$$

7.4. Analysis of quicksort

- Analyze expected running time by recurrence

  Range transformation

$$\mathrm{E}[T(n)] = \frac{n+1}{n}\mathrm{E}[T(n-1)] + \Theta(1)$$

  Define

$$U(n) = \frac{\mathrm{E}[T(n)]}{n+1}$$

  Then,

$$U(n) = \frac{\mathrm{E}[T(n)]}{n+1} = \frac{\mathrm{E}[T(n-1)]}{n} + \Theta\left(\frac{1}{n}\right)$$

$$= U(n-1) + \Theta\left(\frac{1}{n}\right)$$

7.4. Analysis of quicksort

- Analyze expected running time by recurrence

  It is easily seen that

$$U(n) = \sum_{i=1}^{n} \Theta\left(\frac{1}{i}\right) = \Theta\left(\sum_{i=1}^{n} \frac{1}{i}\right) = \Theta(\lg n)$$

  It follows that

$$\begin{aligned} \mathrm{E}[T(n)] &= (n+1)U(n) \\ &= (n+1)\Theta(\lg n) \\ &= \Theta(n \lg n) \end{aligned}$$

7.4. Analysis of quicksort

- Analyze expected running time by recurrence

  Alternative notation

  Instead of defining $T(n)$ as a random variable denoting the running time of Quicksort on n elements, and computing $E[T(n)]$, we may simply define

  $T(n) =$ the *expected* running time of Quicksort on n elements

  and start with

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