6331 - Algorithms, Spring 2014, CSE, OSU
Lecture 3: Quicksort

Instructor: Anastasios Sidiropoulos

January 13, 2014

Sorting

Given an array of integers $A[1 \ldots n]$, rearrange its elements so that

$$A[1] \leq A[2] \leq \ldots \leq A[n].$$

Quicksort

```
Quicksort(A, p, r)
  if p < r
    q = Partition(A, p, r)
    Quicksort(A, p, q − 1)
    Quicksort(A, q + 1, r)
```

Partition

```
Partition(A, p, r)
  x = A[r]
  i = p − 1
  for j = p to r − 1
    if A[j] ≤ x
      i = i + 1
      exchange A[i] with A[j]
  exchange A[i + 1] with A[r]
  return i + 1
```

Partition

```
Partition(A, p, r)
  x = A[r]
  i = p − 1
  for j = p to r − 1
    if A[j] ≤ x
      i = i + 1
      exchange A[i] with A[j]
  exchange A[i + 1] with A[r]
  return i + 1
```

What is the running time of the procedure Partition?

Invariants of Partition

- If $p \leq k \leq i$, then $A[k] \leq x$.
- If $i + 1 \leq k \leq j - 1$, then $A[k] > x$.
- If $k = r$, then $A[k] = x$.

Invariants of Partition

- If $p \leq k \leq i$, then $A[k] \leq x$.
- If $i + 1 \leq k \leq j - 1$, then $A[k] > x$.
- If $k = r$, then $A[k] = x$.

What does the procedure Partition do?

Worst-case performance of Quicksort

Lower bound on the worst-case performance of Quicksort?

Worst-case performance of Quicksort

Lower bound on the worst-case performance of Quicksort?

Unbalanced partition.

Worst-case performance of Quicksort

Lower bound on the worst-case performance of Quicksort?

Unbalanced partition.

$$\begin{aligned} T(n) &\geq T(n-1) + T(0) + \Theta(n) \\ &= T(n-1) + \Theta(n) \\ &= \Omega(n^2) \end{aligned}$$

Worst-case performance of Quicksort

Upper bound on the worst-case performance of Quicksort?

$$T(n) = \max_{0 \leq q \leq n-1} T(q) + T(n-q-1) + \Theta(n)$$

Worst-case performance of Quicksort

Upper bound on the worst-case performance of Quicksort?

$$T(n) = \max_{0 \leq q \leq n-1} T(q) + T(n-q-1) + \Theta(n)$$

We guess $T(n) \leq c \cdot n^2$.

$$\begin{aligned} T(n) &\leq \max_{0 \leq q \leq n-1} (cq^2 + c(n-q-1)^2) + \Theta(n) \\ &= c \cdot \max_{0 \leq q \leq n-1} (q^2 + (n-q-1)^2) + \Theta(n) \end{aligned}$$

Worst-case performance of Quicksort

Upper bound on the worst-case performance of Quicksort?

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

We guess $T(n) \le c \cdot n^2$.

$$\begin{aligned} T(n) &\le \max_{0 \le q \le n-1} (cq^2 + c(n-q-1)^2) + \Theta(n) \\ &= c \cdot \max_{0 \le q \le n-1} (q^2 + (n-q-1)^2) + \Theta(n) \end{aligned}$$

We have $\max_{0 \le q \le n-1}(q^2 + (n-q-1)^2) \le (n-1)^2$.

Worst-case performance of Quicksort

Upper bound on the worst-case performance of Quicksort?

$$T(n) = \max_{0 \leq q \leq n-1} T(q) + T(n-q-1) + \Theta(n)$$

We guess $T(n) \leq c \cdot n^2$.

$$\begin{aligned} T(n) &\leq \max_{0 \leq q \leq n-1} (cq^2 + c(n-q-1)^2) + \Theta(n) \\ &= c \cdot \max_{0 \leq q \leq n-1} (q^2 + (n-q-1)^2) + \Theta(n) \end{aligned}$$

We have $\max_{0 \leq q \leq n-1}(q^2 + (n-q-1)^2) \leq (n-1)^2$.

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

for c a large enough constant. Thus, $T(n) = \Theta(n^2)$.

Performance of Quicksort

What happens when all the elements of A are equal?

Performance of Quicksort

What happens when all the elements of A are equal?

For the rest of the lecture, we will assume that all elements are distinct.

Randomized Quicksort

Pick the pivot *randomly*.

Randomized Quicksort

Pick the pivot *randomly*.

Why would that make any difference?

Randomized Quicksort

Pick the pivot *randomly*.

Why would that make any difference?

Is this the same as average-case analysis?

Randomized algorithms vs random input

The average running time on an algorithm for some **distribution** over inputs, analysis is *not* the same as the expected running time of a randomized algorithm over an **arbitrary** input.

Randomized algorithms vs random input

The average running time on an algorithm for some **distribution** over inputs, analysis is *not* the same as the expected running time of a randomized algorithm over an **arbitrary** input.

Examples?

Randomized Quicksort

Randomized-Partition(A, p, r)
  $i =$ Random(p, r)
  exchange $A[i]$ with $A[i]$
  return Partition(A, p, r)

Randomized Quicksort

Randomized-Partition(A, p, r)
  $i =$ Random(p, r)
  exchange $A[i]$ 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$(A, q + 1, r)$

Randomized Quicksort

Randomized-Partition(A, p, r)
  $i =$ Random(p, r)
  exchange $A[i]$ 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($A, q + 1, r$)

What is the running time of the procedure Randomized-Partition?

Expected running time of Randomized-Quicksort

Suppose the elements in A are $z_1, \ldots, z_n$, with

$$z_1 < z_2 < \ldots < z_n.$$

Expected running time of Randomized-Quicksort

The running time is dominated by the number of comparisons.

Expected running time of Randomized-Quicksort

The running time is dominated by the number of comparisons.
Consider the indicator variable

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

Expected running time of Randomized-Quicksort

The running time is dominated by the number of comparisons. Consider the indicator variable

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

The total number of comparisons is

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

Expected running time of Randomized-Quicksort

The expected running time is

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

The probability of a comparison

Suppose $i < j$.

$$\begin{aligned}
\Pr\{z_i \text{ is compared to } z_j\} &= \Pr\{z_i \text{ or } z_j \text{ is the first pivot in } \{z_i, \ldots, z_j\}\} \\
&\le \Pr\{z_i \text{ is the first pivot in } \{z_i, \ldots, z_j\}\} \\
&\quad + \Pr\{z_j \text{ is the first pivot in } \{z_i, \ldots, z_j\}\} \\
&= \frac{1}{j - i + 1} + \frac{1}{j - 1 + 1} \\
&= \frac{2}{j - i + 1}
\end{aligned}$$

Expected running time of Randomized-Quicksort

The expected running time is

$$\begin{aligned} E[X] &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \Pr\{z_i \text{ is compared to } z_j\} \\ &\leq \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} \\ &\leq \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1} \\ &= O(n \cdot \log n) \end{aligned}$$