Analysis of Approximate Quickselect and Related Problems

Conrado Martínez

Univ. Politècnica Catalunya

Joint work with A. Panholzer and H. Prodinger

LIP6, Paris, April 2009

Introduction

- Quickselect finds the kth smallest element out of n given elements with average cost $\Theta(n)$
- Approximate Quickselect selects, out of n given elements, an element whose rank k fails within a prespecified rank $[i..j]$
- We analyze the exact number of passes (recursive calls) and number of key comparisons made by AQS, for arbitrary i, j and n
- Asymptotic estimates follow easily from the exact results

Introduction

- Quickselect finds the kth smallest element out of n given elements with average cost $\Theta(n)$
- Approximate Quickselect selects, out of n given elements, an element whose rank k fails within a prespecified rank $[i..j]$
- We analyze the exact number of passes (recursive calls) and number of key comparisons made by AQS, for arbitrary i, j and n
- Asymptotic estimates follow easily from the exact results

Introduction

- Quickselect finds the kth smallest element out of n given elements with average cost $\Theta(n)$
- Approximate Quickselect selects, out of n given elements, an element whose rank k fails within a prespecified rank $[i..j]$
- We analyze the exact number of passes (recursive calls) and number of key comparisons made by AQS, for arbitrary i, j and n
- Asymptotic estimates follow easily from the exact results

Introduction

- Quickselect finds the kth smallest element out of n given elements with average cost $\Theta(n)$
- Approximate Quickselect selects, out of n given elements, an element whose rank k fails within a prespecified rank $[i..j]$
- We analyze the exact number of passes (recursive calls) and number of key comparisons made by AQS, for arbitrary i, j and n
- Asymptotic estimates follow easily from the exact results

Introduction

- The techniques we use to analyze Approximate Quickselect prove useful to analyze other problems as well
  1. The number of moves in which the ith element gets involved when selecting the jth smallest element out of n
  2. The number of common ancestors of the nodes of rank i and j in a random binary search tree (BST) of size n
  3. The size of the smallest subtree containing the nodes i and j in a random BST of size n
  4. The distance (number of edges) between the nodes i and j in a random BST of size n
- We analyze also Approximate Multiple Quickselect, an algorithm to find q elements with ranks falling in prespecified ranges $[i_1..j_1], [i_2..j_2], \ldots, [i_q..j_q]$

Introduction

- The techniques we use to analyze Approximate Quickselect prove useful to analyze other problems as well
  1. The number of moves in which the ith element gets involved when selecting the jth smallest element out of n
  2. The number of common ancestors of the nodes of rank i and j in a random binary search tree (BST) of size n
  3. The size of the smallest subtree containing the nodes i and j in a random BST of size n
  4. The distance (number of edges) between the nodes i and j in a random BST of size n
- We analyze also Approximate Multiple Quickselect, an algorithm to find q elements with ranks falling in prespecified ranges $[i_1..j_1], [i_2..j_2], \ldots, [i_q..j_q]$

Introduction

- The techniques we use to analyze Approximate Quickselect prove useful to analyze other problems as well
  1. The number of moves in which the ith element gets involved when selecting the jth smallest element out of n
  2. The number of common ancestors of the nodes of rank i and j in a random binary search tree (BST) of size n
  3. The size of the smallest subtree containing the nodes i and j in a random BST of size n
  4. The distance (number of edges) between the nodes i and j in a random BST of size n
- We analyze also Approximate Multiple Quickselect, an algorithm to find q elements with ranks failing in prespecified ranges $[i_1..j_1], [i_2..j_2], \ldots, [i_q..j_q]$

Introduction

- The techniques we use to analyze Approximate Quickselect prove useful to analyze other problems as well
  1. The number of moves in which the ith element gets involved when selecting the jth smallest element out of n
  2. The number of common ancestors of the nodes of rank i and j in a random binary search tree (BST) of size n
  3. The size of the smallest subtree containing the nodes i and j in a random BST of size n
  4. The distance (number of edges) between the nodes i and j in a random BST of size n
- We analyze also Approximate Multiple Quickselect, an algorithm to find q elements with ranks failing in prespecified ranges $[i_1..j_1], [i_2..j_2], \ldots, [i_q..j_q]$

Introduction

- The techniques we use to analyze Approximate Quickselect prove useful to analyze other problems as well
  1. The number of moves in which the ith element gets involved when selecting the jth smallest element out of n
  2. The number of common ancestors of the nodes of rank i and j in a random binary search tree (BST) of size n
  3. The size of the smallest subtree containing the nodes i and j in a random BST of size n
  4. The distance (number of edges) between the nodes i and j in a random BST of size n
- We analyze also Approximate Multiple Quickselect, an algorithm to find q elements with ranks failing in prespecified ranges $[i_1..j_1], [i_2..j_2], \ldots, [i_q..j_q]$

Introduction

- The techniques we use to analyze Approximate Quickselect prove useful to analyze other problems as well
  1. The number of moves in which the ith element gets involved when selecting the jth smallest element out of n
  2. The number of common ancestors of the nodes of rank i and j in a random binary search tree (BST) of size n
  3. The size of the smallest subtree containing the nodes i and j in a random BST of size n
  4. The distance (number of edges) between the nodes i and j in a random BST of size n
- We analyze also Approximate Multiple Quickselect, an algorithm to find q elements with ranks failing in prespecified ranges $[i_1..j_1], [i_2..j_2], \ldots, [i_q..j_q]$

The algorithm

Ensure: Array $A[l..r]$, integers i and j with $l \leq i \leq j \leq r$
Require: Returns a value k, with $i \leq k \leq j$, $A[k]$ has rank between $i - l + 1$ and $j - l + 1$ in the array $A[l..r]$

procedure AQS(A, i, j, l, r)
  if $r - l \leq j - i$ **then return** l
  end if
  PARTITION(A, l, r, k)
  $\{ \forall m : (l \leq m < k) \Rightarrow A[m] \leq A[k]$, and
    $\forall m : (k < m \leq r) \Rightarrow A[k] \leq A[m] \}$
  if $j < k$ **then return** AQS($A, i, j, l, k - 1$)
  else if $i > k$ **then return** AQS($A, i, j, k + 1, r$)
  else return k
  end if
end procedure

The number of passes

$P_{n,i,j}$ = the average number of recursive calls to AQS to select an element of rank $k \in [i..j]$ out of n

$$P_{n,i,j} = 1 + \frac{1}{n}\sum_{k=1}^{i-1} P_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} P_{k-1,i,j}, \quad \text{for } 1 \leq i \leq j \leq n$$

The number of passes

$P_{n,i,j}$ = the average number of recursive calls to AQS to select an element of rank $k \in [i..j]$ out of n

$$P_{n,i,j} = 1 + \frac{1}{n}\sum_{k=1}^{i-1} P_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} P_{k-1,i,j}, \quad \text{for } 1 \le i \le j \le n$$

The initial recursive call

The number of passes

$P_{n,i,j}$ = the average number of recursive calls to AQS to select an element of rank $k \in [i..j]$ out of n

$$P_{n,i,j} = 1 + \frac{1}{n}\sum_{k=1}^{i-1} P_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} P_{k-1,i,j}, \quad \text{for } 1 \le i \le j \le n$$

If the pivot lands at $k < i$ we continue in the right subarray of $n - k$ elements looking for an element with rank in $[i - k..j - k]$

The number of passes

$P_{n,i,j}$ = the average number of recursive calls to AQS to select an element of rank $k \in [i..j]$ out of n

$$P_{n,i,j} = 1 + \frac{1}{n}\sum_{k=1}^{i-1} P_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} P_{k-1,i,j}, \quad \text{for } 1 \le i \le j \le n$$

If the pivot lands at $k > j$ we continue in the left subarray of $k-1$ elements looking for an element with rank in $[i..j]$

The number of passes

$P_{n,i,j}$ = the average number of recursive calls to AQS to select an element of rank $k \in [i..j]$ out of n

$$P_{n,i,j} = 1 + \frac{1}{n}\sum_{k=1}^{i-1} P_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} P_{k-1,i,j}, \quad \text{for } 1 \leq i \leq j \leq n$$

If the pivots lands at k, $i \leq k \leq j$, we are done

The number of comparisons

$C_{n,i,j}$ = the average number of key comparisons in AQS to select an element of rank $k \in [i..j]$ out of n

$$C_{n,i,j} = n-1 + \frac{1}{n} \sum_{k=1}^{i-1} C_{n-k,i-k,j-k} + \frac{1}{n} \sum_{k=j+1}^{n} C_{k-1,i,j}, \quad \text{for } 1 \le i \le j \le n$$

The generic trivariate recurrence

$T_{n,i,j}$ = generic "toll" function

$$X_{n,i,j} = T_{n,i,j} + \frac{1}{n} \sum_{k=1}^{i-1} X_{n-k,i-k,j-k} + \frac{1}{n} \sum_{k=j+1}^{n} X_{k-1,i,j}, \quad \text{for } 1 \le i \le j \le n$$

Example

- $T_{n,i,j} = 1 \Rightarrow$ passes
- $T_{n,i,j} = n - 1 \Rightarrow$ comparisons
- $T_{n,i,j} = \frac{n}{6} + O(1) \Rightarrow$ swaps
- $i = j \Rightarrow$ Quickselect

The generic trivariate recurrence

- Define:

$$X(z,u_1,u_2) := \sum_{i\geq 1}\sum_{j\geq i}\sum_{n\geq j} X_{n,i,j} z^n u_1^i u_2^j,$$

$$T(z,u_1,u_2) := \sum_{i\geq 1}\sum_{j\geq i}\sum_{n\geq j} T_{n,i,j} z^n u_1^i u_2^j.$$

- The trivariate recurrence translates to

$$\frac{\partial}{\partial z} X(z,u_1,u_2) = \left(\frac{1}{1-z} + \frac{u_1 u_2}{1-zu_1u_2}\right) X(z,u_1,u_2) + \frac{\partial}{\partial z} T(z,u_1,u$$

with initial condition $X(0,u_1,u_2) = 0$.

The generic trivariate recurrence

- Define:

$$X(z,u_1,u_2) := \sum_{i\geq 1}\sum_{j\geq i}\sum_{n\geq j} X_{n,i,j} z^n u_1^i u_2^j,$$

$$T(z,u_1,u_2) := \sum_{i\geq 1}\sum_{j\geq i}\sum_{n\geq j} T_{n,i,j} z^n u_1^i u_2^j.$$

- The trivariate recurrence translates to

$$\frac{\partial}{\partial z}X(z,u_1,u_2) = \left(\frac{1}{1-z} + \frac{u_1u_2}{1-zu_1u_2}\right)X(z,u_1,u_2) + \frac{\partial}{\partial z}T(z,u_1,u$$

with initial condition $X(0,u_1,u_2) = 0$.

The generic trivariate recurrence

Lemma

$$X(z,u_1,u_2) = \frac{1}{(1-z)(1-zu_1u_2)} \times \int_0^z (1-t)(1-u_1u_2t)\Big(\frac{\partial}{\partial t}T(t,u_1,u_2)\Big)dt.$$

The generic trivariate recurrence

Theorem

$$X_{n,i,j} = \sum_{\ell=1}^{i-1} \sum_{k=j-i+\ell}^{n-i+\ell-1} \frac{2T_{k,\ell,j-i+\ell}}{(k+1)(k+2)} + \sum_{\ell=1}^{i-1} \frac{T_{n-i+\ell,\ell,j-i+\ell}}{n-i+\ell+1} + \sum_{k=j}^{n-1} \frac{T_{k,i,j}}{k+1} + T_{n,i,j}$$

Setting $i = j$ we rederive the generic solution for Quickselect-like recurrences by Kuba (2006).

Analyzing AQS

Theorem

The expected number of passes in AQS is*

$$P_{n,i,j} = H_j + H_{n-i+1} - 2H_{j-i+1} + 1.$$

The expected number of comparisons in AQS is

$$\begin{aligned} C_{n,i,j} = 2(n+1)H_n + 2(j-i+4)H_{j-i+1} - 2(j+2)H_j \\ - 2(n-i+3)H_{n-i+1} + 2n - j + i - 2. \end{aligned}$$

$$(^*) \quad H_n := \sum_{1 \le k \le n} \frac{1}{k}, \quad n\text{th harmonic number}$$

Moves in Quickselect

Consider the ith smallest element in the array $A[1..n]$. How many times do we move it around when selecting the jth smallest element in A?

Moves in Quickselect

We make a case analysis, comparing with the position k where the pivot lands after a partitioning step:

- if $k < i \leq j$ or $k < j \leq i$, Quickselect continues recursively in $A[k + 1..n]$ that does contain ith element
- if $j \leq i < k$ or $i \leq j < k$, Quickselect continues in $A[1..k - 1]$ that does contain the ith element
- if $i \leq k \leq j$ or $j \leq k \leq i$, either Quickselect stops or continues in a subarray not containing ith element

Moves in Quickselect

We make a case analysis, comparing with the position k where the pivot lands after a partitioning step:

- if $k < i \leq j$ or $k < j \leq i$, Quickselect continues recursively in $A[k+1..n]$ that does contain ith element
- if $j \leq i < k$ or $i \leq j < k$, Quickselect continues in $A[1..k-1]$ that does contain the ith element
- if $i \leq k \leq j$ or $j \leq k \leq i$, either Quickselect stops or continues in a subarray not containing ith element

Moves in Quickselect

We make a case analysis, comparing with the position k where the pivot lands after a partitioning step:

- if $k < i \leq j$ or $k < j \leq i$, Quickselect continues recursively in $A[k+1..n]$ that does contain ith element
- if $j \leq i < k$ or $i \leq j < k$, Quickselect continues in $A[1..k-1]$ that does contain the ith element
- if $i \leq k \leq j$ or $j \leq k \leq i$, either Quickselect stops or continues in a subarray not containing ith element

Moves in Quickselect

To find the toll function (number of moves where i participates) in a single partitioning step, we also consider three cases:

1. $i = k \Rightarrow$ the element i is moved (once)
2. $i < k \Rightarrow$ the element i is moved if it were in $A[k..n] \Rightarrow$ prob $= (n - k + 1)/(n - 1)$
3. $i > k \Rightarrow$ the element is moved if it were in $A[2..k] \Rightarrow$ prob $= (k - 1)/(n - 1)$

Moves in Quickselect

$M_{n,i,j}$:= The expected number of moves of the ith element when selecting the jth smallest out of n

$$M_{n,i,j} = \frac{1}{n}\sum_{k=1}^{i-1} M_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} M_{k-1,i,j} + \frac{(i-1)(i-2)}{2n(n-1)} + \frac{(n-i)(n-i+1)}{2n(n-1)} + \frac{1}{n}, \quad 1 \leq i < j \leq n$$

Analogous recurrences for $i = j$ and $j < i$, all three solved using the theorem for trivariate recurrences

Moves in Quickselect

$M_{n,i,j}$:= The expected number of moves of the ith element when selecting the jth smallest out of n

$$M_{n,i,j} = \frac{1}{n}\sum_{k=1}^{i-1} M_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} M_{k-1,i,j} + \frac{(i-1)(i-2)}{2n(n-1)} + \frac{(n-i)(n-i+1)}{2n(n-1)} + \frac{1}{n}, \quad 1 \le i < j \le n$$

Analogous recurrences for $i = j$ and $j < i$, all three solved using the theorem for trivariate recurrences

Binary search trees

- $A_{n,i,j}$:= average # of common ancestors of nodes i and j
- $S_{n,i,j}$:= average size of smallest subtree containing nodes i and j
- $D_{n,i,j}$:= average distance (# of edges) between nodes i and j

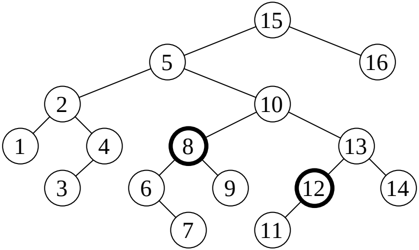
Binary search trees

- $A_{n,i,j}$:= average # of common ancestors of nodes i and j
- $S_{n,i,j}$:= average size of smallest subtree containing nodes i and j
- $D_{n,i,j}$:= average distance (# of edges) between nodes i and j

Binary search trees

- $A_{n,i,j}$:= average # of common ancestors of nodes i and j
- $S_{n,i,j}$:= average size of smallest subtree containing nodes i and j
- $D_{n,i,j}$:= average distance (# of edges) between nodes i and j

Binary search trees



Example

$$A_{16,8,12} = 3 \qquad \text{(nodes 15, 5, 10)}$$
$$S_{16,8,12} = 9 \qquad \text{(subtree rooted at 10)}$$
$$D_{16,8,12} = 3$$

Binary search trees

if $T = \circ(L, R)$ is random binary search tree (BST) of size $n > 0$, then

1. Any element has identical probability of being the root, thus

$$\Pr\{|L| = k - 1 \,|\, |T| = n\} = \frac{1}{n}, \qquad 1 \le k \le n$$

2. L and R are independent random BSTs of sizes $k - 1$ and $n - k$, $1 \le k \le n$

Binary search trees

if $T = \circ(L, R)$ is random binary search tree (BST) of size $n > 0$, then

1. Any element has identical probability of being the root, thus

$$\Pr\{|L| = k - 1 \,|\, |T| = n\} = \frac{1}{n}, \qquad 1 \leq k \leq n$$

2. L and R are independent random BSTs of sizes $k - 1$ and $n - k$, $1 \leq k \leq n$

Common ancestors

Suppose that the root of the BST is the kth element.

- if $i \leq j < k$ the number of common ancestors is 1 + the number of common ancestors in a random of BST of size $k - 1$
- if $k < i \leq j$ the number of common ancestors is 1 + the number of common ancestors in a random of BST of size $n - 1 - k$ (of the nodes of ranks $i - k$ and $j - k$!)
- if $i \leq k \leq j$, then there is only one common ancestor (k)

Lemma

$$A_{n,i,j} = P_{n,i,j} = passes\ in\ AQS$$

Common ancestors

Suppose that the root of the BST is the kth element.

- if $i \leq j < k$ the number of common ancestors is 1 + the number of common ancestors in a random of BST of size $k - 1$
- if $k < i \leq j$ the number of common ancestors is 1 + the number of common ancestors in a random of BST of size $n - 1 - k$ (of the nodes of ranks $i - k$ and $j - k$!)
- if $i \leq k \leq j$, then there is only one common ancestor (k)

Lemma

$$A_{n,i,j} = P_{n,i,j} = \textit{passes in AQS}$$

Common ancestors

Suppose that the root of the BST is the kth element.

- if $i \leq j < k$ the number of common ancestors is 1 + the number of common ancestors in a random of BST of size $k - 1$
- if $k < i \leq j$ the number of common ancestors is 1 + the number of common ancestors in a random of BST of size $n - 1 - k$ (of the nodes of ranks $i - k$ and $j - k$!)
- if $i \leq k \leq j$, then there is only one common ancestor (k)

Lemma

$$A_{n,i,j} = P_{n,i,j} = passes\ in\ AQS$$

Common ancestors

Suppose that the root of the BST is the kth element.

- if $i \leq j < k$ the number of common ancestors is 1 + the number of common ancestors in a random of BST of size $k - 1$
- if $k < i \leq j$ the number of common ancestors is 1 + the number of common ancestors in a random of BST of size $n - 1 - k$ (of the nodes of ranks $i - k$ and $j - k$!)
- if $i \leq k \leq j$, then there is only one common ancestor (k)

Lemma

$$A_{n,i,j} = P_{n,i,j} = \textit{passes in AQS}$$

Size of subtree of LCA

The recurrence for $S_{n,i,j}$ follows the usual pattern:

$$\begin{aligned} S_{n,i,j} &= \frac{1}{n}\sum_{k=1}^{i-1} S_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} S_{k-1,i,j} + \frac{1}{n}\sum_{k=i}^{j} n \\ &= \frac{1}{n}\sum_{k=1}^{i-1} S_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} S_{k-1,i,j} + j - i + 1 \end{aligned}$$

Size of subtree of LCA

The recurrence for $S_{n,i,j}$ follows the usual pattern:

$$
\begin{aligned}
S_{n,i,j} &= \frac{1}{n}\sum_{k=1}^{i-1} S_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} S_{k-1,i,j} + \frac{1}{n}\sum_{k=i}^{j} n \\
&= \frac{1}{n}\sum_{k=1}^{i-1} S_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} S_{k-1,i,j} + j - i + 1
\end{aligned}
$$

The toll function is n only when the pivot k satisfies $i \leq k \leq j$

Size of subtree of LCA

Apply the theorem for trivariate recurrences with $T_{n,i,j} := j - i + 1$

Theorem

$$
\begin{aligned}
S_{n,i,j} &= (j-i+1)\bigl(H_j + H_{n-i+1} - 2H_{j-i+1} + 1\bigr) = (j-i+1)\cdot A_{n,i,j} \\
&\approx (j-i+1)\bigl(\log j + \log(n-i+1) - 2\log(j-i+1) + 1\bigr)
\end{aligned}
$$

Distance

The recurrence for $D_{n,i,j}$:

$$D_{n,i,j} = \frac{1}{n}\sum_{k=1}^{i-1} D_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} D_{k-1,i,j} + \frac{1}{n}\sum_{k=i}^{j}(A_{k-1,i,i} + A_{n-k,j-k,j-k} + 2)$$

Distance

The recurrence for $D_{n,i,j}$:

$$D_{n,i,j} = \frac{1}{n}\sum_{k=1}^{i-1} D_{n-k,i-k,j-k} + \frac{1}{n}\sum_{k=j+1}^{n} D_{k-1,i,j} + \frac{1}{n}\sum_{k=i}^{j}(A_{k-1,i,i} + A_{n-k,j-k,j-k} + 2)$$

If k is the LCA of i and j, the distance between i and j is the depth of i in the left subtree ($A_{k-1,i,i}$), plus the depth of j (the $(j-k)$th element) in the right subtree ($A_{n-k,j-k,j-k}$), plus 2

Distance

Since we know $A_{n,i,j}$, we can obtain the toll function for distances

$$\frac{1}{n}\sum_{k=i}^{j}(A_{k-1,i,i} + A_{n-k,j-k,j-k} + 2)$$
$$= \frac{j-i+1}{n}\left(H_i + H_{n+1-j} + 2H_{j+1-i} - 2\right)$$

Distance

The last step is to apply the theorem of trivariate recurrences with the toll function above (quite laboriously!)

Theorem

$$D_{n,i,j} = 4H_{j+1-i} - (H_j - H_i) - (H_{n+1-i} - H_{n+1-j}) - 3$$