Expected Time Bounds for Selection

Λέα Τζανής
ΑΛΜΑ
Δεκέμβριος,2021

Introduction

selection problem can be succinctly stated as follows: given a set X of n distinct numbers and an integer i, $1 \leq i \leq n$, determine the ith smallest element of X with as few comparisons as possible. The ith smallest element, denoted by $i \theta X$, is that element which is larger than exactly $i - 1$ other elements, so that $1 \theta X$ is the smallest, and $n \theta X$ the largest, element in X.

Definitions

- $f(i,n)$ = "The expected number of comparisons required to select $i\theta X$"
- $t\rho X$ = "Rank of an element $t \in X$, so that $(t\rho X)\theta X=t$"

Algorithms

1. Hoare's Find (Quickselect)
2. Select (Version 1)
3. Improved Select (Version 2)

Hoare's Find/Quickselect

Given an array A[l,…,p], we search for the ith smallest element of the array

- Select (random) a pivot element
- Partition (A[l,…,q],A[q+1,…,p])
- Compute the index k of pivot element
  1. if k=i, then A[k] is the answer
  2. If k>i, then run quickselect for A[l,…,q]
  3. If k<I, then run quickselect for A[q+1,p]

Trivial Lower and Upper Bound

$$f(i,n) \geq n - 1, \quad \text{for} \quad 1 \leq i \leq n. \qquad (1)$$

For every selection algorithm

$$f(i,n) \leq 2((n+1)H_n - (n+3-i)H_{n-i+1} - (i+2)H_i + n + 3), \qquad (2)$$

THE ANALYSIS OF RANGE QUICKSELECT AND RELATED PROBLEMS
CONRADO MART´INEZ, ALOIS PANHOLZER, AND HELMUT PRODINGER

Select

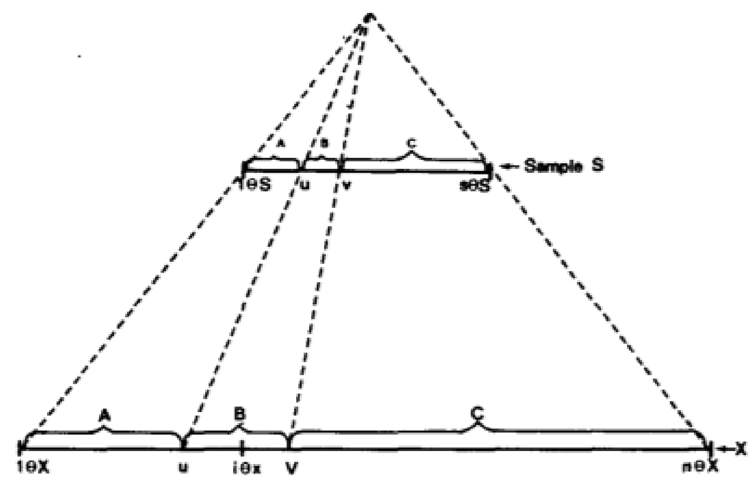
Step 1. A small random sample S of size $s = s(n)$ is drawn from X.

Step 2. Two elements, u and v, $(u < v)$, are selected from S, using *SELECT* recursively, such that the set $\{x \in X \mid u \leq x \leq v\}$ is expected to be of size $o(n)$ and yet expected to contain $i \theta X$. Selecting u and v partitions S into those elements less than u (set A), those elements between u and v (set B), and those elements greater than v (set C).

Step 3. The partitioning of X into these three sets is then completed by comparing each element x in $X - S$ to u and v. If $i < \lceil n/2 \rceil$, x is compared to v first, and then to u only if $x < v$. If $i \geq \lceil n/2 \rceil$, the order of the comparisons is reversed.

Step 4. With probability approaching 1 (as $n \to \infty$), $i \theta X$ will lie in set B, and the algorithm is applied recursively to select $i \theta X$ from B. (Otherwise *SELECT* is applied to A or C as appropriate.)

Fig. 1.



Total comparisons

- Choice of u and v
- Partitioning (both elements of S and X\S)
- Select iθX from B
- Select iθX from A or C
- f(i,n) = sum(all of the above)

If $s(n)$, u, and v can be chosen so that $s(n) = o(n)$, $E(\,|B|\,) = o(n)$, and $P(i\,\theta\,X \notin B) = o(n^{-1})$, then the total number of comparisons expected is:

$$n + \min(i, n - i) + o(n)$$

Choice of u and v

For fixed tρS, we can compute where t should fall in X (Expected value and variance)

$$E(t\,\rho\,X) = \frac{(n+1)}{(s+1)}(t\,\rho\,S), \qquad (10)$$

$$\sigma(t\,\rho\,X) = \left(\frac{(t\,\rho\,S)(s-(t\,\rho\,S)+1)(n+1)(n-s)}{(s+1)^2(s+2)}\right)^{\frac{1}{2}}, \qquad (11)$$
$$\leq \frac{1}{2}\left(\frac{(n+1)(n-s)}{s}\right)^{\frac{1}{2}} \leq \frac{1}{2}\frac{n}{(s)^{\frac{1}{2}}}.$$

For the conditions reported previously

$$E(u\,\rho\,X) + 2d\sigma(u\,\rho\,X) \cong i \cong E(v\,\rho\,X) - 2d\sigma(v\,\rho\,X), \qquad (12)$$

where d=d(n) a slowly unbounded growing function of n
(d=(ln(n))^(1/2) to ensure $P(i < u\,\rho\,X \text{ or } i > v\,\rho\,X) = o(n^{-1})$)

The above equations mean that

$$u\,\rho\,S \cong \left(i - d\left(\frac{(n+1)(n-s)}{s}\right)^{\frac{1}{2}}\right)\left(\frac{s+1}{n+1}\right) \geq \frac{i(s+1)}{(n+1)} - d(s)^{\frac{1}{2}},$$

and

$$v\,\rho\,S \cong \left(i + d\left(\frac{(n+1)(n-s)}{s}\right)^{\frac{1}{2}}\right)\left(\frac{s+1}{n+1}\right) \leq \frac{i(s+1)}{(n+1)} + d(s)^{\frac{1}{2}}. \qquad (14)$$

Upper bound for select

Let $g(i,n)$ denote the expected number of comparisons made by *SELECT*. It will be shown inductively that

$$g(i,n) = n + \min(i,n-i) + O(n^{2/3} \ln^{1/3}(n)) \qquad (15)$$

The cost of selecting u and v can be estimated as follows:

- First we apply select recursively to S to select u, and then
- We extract v from those elements of S which are greater than u.

These to operations cost:

$$\begin{aligned} &g(u \,\rho\, S, s) + g(v \,\rho\, S - u \,\rho\, S + 1, s - u \,\rho\, S) \\ &\quad \le 2s + v \,\rho\, S - u \,\rho\, S + O(s^{2/3} \ln^{1/3}(s)) \qquad (16) \\ &\quad \le 2s + 2d(s)^{1/2} + O(s^{2/3} \ln^{1/3}(s)) \end{aligned}$$

comparisons.

There are n-s(n) elements to compare, and the propability that 2 comparisons will be made for an element is *min(uρS,s+1-uρS)/(s+1)* so that the total is:

$$(n - s(n))(1 + \min(i,n-i)/n + ds^{-1/2}). \qquad (17)$$

The cost of finishing up, if $i \,\theta\, X$ falls in B, is at most $g(|B|/2, |B|)$. But

$$E(|B|) = (v \,\rho\, S - u \,\rho\, S)n/s = 2dns^{-1/2} \qquad (18)$$

so that

$$g(|B|/2, |B|) = 3dns^{-1/2} + O((dns^{-1/2})^{2/3}(\ln(dns^{-1/2}))^{1/3}). \qquad (19)$$

Upper Bound for Select

Considering that the probability that iθXϵA or ιθXϵC is from (13) less than c/(dn), so that the Total work expected in this case is less than 3c/(2d) (which goes to 0 as n-->∞) we have the total cost :

$$
\begin{aligned}
g(i,n) \le\ & 2s + 2d(s)^{\frac{1}{2}} + O(s^{\frac{2}{3}} \ln^{\frac{1}{3}}(s)) \\
& + (n - s)(1 + \min(i,n - i)/n + ds^{-\frac{1}{2}}) \\
& + 2dns^{-\frac{1}{2}} + 3c/(2d) \\
\le\ & n + \min(i,n - i) + s + ds^{\frac{1}{2}} \\
& - \min(i,n - i)s/n \\
& + 3dns^{-\frac{1}{2}} + 3c/(2d) + O(s^{\frac{2}{3}} \ln^{\frac{1}{3}} s). \qquad (20)
\end{aligned}
$$

Improved Select

Let $S_1 \subset S_2 \subset \cdots \subset S_k = X$ be a nested series of random samples from X of sizes $s_1, s_2, \cdots, s_k = n$. For each sample S_j, let u_j and v_j be chosen from S_j as in (14) so that

$$u_j \,\rho\, S_j = \left(i - d\left(\frac{(n+1)(n-s_j)}{s_j}\right)^{\frac{1}{2}}\right) \cdot \left(\frac{s_j+1}{n+1}\right)$$

and (22)

$$v_j \,\rho\, S_j = \left(i + d\left(\frac{(n+1)(n-s_j)}{s_j}\right)^{\frac{1}{2}}\right) \cdot \left(\frac{s_j+1}{n+1}\right).$$

Thus it is very likely, for any j, that $u_j \,\rho\, X \leq i \leq v_j \,\rho\, X$. Furthermore, as j approaches k (i.e. as s_j gets large), u_j and v_j surround $i \,\theta\, X$ ever more closely. In fact, $u_k = i \,\theta\, X = v_k$. The cost of finding u_j and v_j directly from S_j is of course prohibitive for large values of s_j. However, since

$$E(u_{j-1} \,\rho\, S_j) = (u_{j-1} \,\rho\, S_{j-1}) \cdot \frac{s_j + 1}{s_{j-1} + 1} \leq u_j \,\rho\, S_j, \quad (23)$$

and similarly $E(v_{j-1} \,\rho\, S_j) \geq v_j \,\rho\, S_j$, we can use u_{j-1} and v_{j-1} to bound the search for u_j and v_j.

Improved Select

Step 1. Draw a random sample S_1 of size s_1 from X, and select u_1 and v_1 using this algorithm recursively (and the ranks given in (22).

Step 2. Determine the sets A_2, B_2, and C_2, a partition of S_2, by comparing each element in $S_2 - S_1$ to u_1 and v_1 (using the same order of comparison strategy as the original *SELECT*).

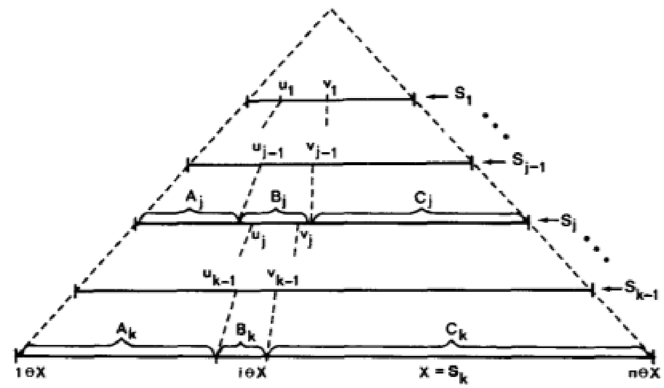
Step 3. Next, determine u_2 and v_2 by applying this algorithm recursively to B_2 (in the most likely case; else A_2 or C_2).

Step 4. Extend the partition of S_2 determined by u_2 and v_2 into a partition A_3, B_3, C_3 of S_3 by comparing each element of $S_3 - S_2$ to u_2 and v_2 with the same comparison strategy.

Step 5. Continue in this fashion until a partition A_k, B_k, C_k of the set $S_k = X$ has been created.

Step 6. Then use the algorithm recursively once more to extract $i \theta X$ from B_k (or A_k or C_k, if necessary).

Fig. 2.



Upper Bound for Select (version 2)

$$g(j,m) = m + \min(j,m-j) + O(m^{1/2}), \quad \text{for} \quad m < n,\ 1 \le j \le m. \tag{26}$$

The expected size of B_j is easily estimated:

$$E(|B_j|) = (v_{j-1}\,\rho\, S_{j-1} - u_{j-1}\,\rho\, S_{j-1}) \cdot \left(\frac{s_j}{s_{j-1}}\right) \le 2ds_j/(s_{j-1})^{1/2}. \tag{27}$$

The cost of selecting $u_2, v_2, \ldots, u_{k-1}, v_{k-1}$ from the sets $B_2, \ldots, B_{k-1}$ is just

$$\sum_{2\le j\le k-1} (g(u_j\,\rho\, B_j, |B_j|) + g(v_j\,\rho\, B_j - u_j\,\rho\, B_j + 1, |B_j| - u_j\,\rho\, B_j)) \le \sum_{2\le j\le k-1} (4ds_j/(s_{j-1})^{1/2} + 2d(s_j)^{1/2}), \tag{28}$$

The cost of partitioning $S_2 - S_1, S_3 - S_2, \ldots, S_k - S_{k-1}$ about u_1 and v_1, u_2 and $v_2, \ldots, u_{k-1}$ and v_{k-1} is just

$$\sum_{2\le j\le k} (s_j - s_{j-1})(1 + \min(i,n-i)/n + d/(s_{j-1})^{1/2}). \tag{31}$$

Adding these all together, we have

$$\begin{aligned} g(i,n) \le n + \min(i,n-i) &+ \sum_{2\le j\le k} (5ds_j/(s_{j-1})^{1/2} + d(s_j)^{1/2}) \\ &+ s_1(1 - \min(i,n-i)/n) + d(s_1)^{1/2} \\ &- dn/(s_{k-1})^{1/2}. \end{aligned} \tag{32}$$

This sum can be approximately minimized if we let $s_1, s_2, \ldots, s_k$ increase geometrically with ratio r^2, so that $s_j = r^{2j-2}s_1$, and

$$\begin{aligned} g(i,n) &\le n + \min(i,n-i) + \left(\frac{5d}{(s_1)^{1/2}} + \frac{(s_1)^{1/2}}{r}\right) \cdot \sum_{2\le j\le k} r^j \\ &\le n + \min(i,n-i) + \left(\frac{5d}{(s_1)^{1/2}} + \frac{(s_1)^{1/2}}{r}\right) \cdot \left(\frac{r^{k-1}-1}{r-1}\right) \cdot r^2 \\ &\le n + \min(i,n-i) + (n)^{1/2}\left(\frac{r^2}{r-1}\right)\left(\frac{5d}{s_1} + \frac{1}{r}\right). \end{aligned} \tag{33}$$

This is approximately minimized when $s_1 = \ln^{1/2} n$, and $r = 4.32$, yielding

$$g(i,n) \le n + \min(i,n-i) + O(n^{1/2}), \tag{34}$$

Where the cost of selecting u1,v1 from S1 < 2s1 + 2(d(s1))^1/2, and the cost of selecting uk=vk=iθX from Bk is (3dn)/(sk-1)^(1/2)

On the lower bound of (n-1) comparisons

THEOREM 1. *Any selection algorithm that has determined $i \theta X$ to be some element $y \in X$ must also have determined, for any $x \in X$, $x \neq y$, whether $x < y$ or $y < x$.*

PROOF. Assume that there exists an x incomparable with y in the partial order determined by the algorithm. Then there exists a linear ordering of X, consistent with the partial order determined, in which x and y are adjacent (since any element required to lie between x and y would imply a relationship between x and y in the partial order). But then x and y may be interchanged in the linear order without contradicting the partial order—demonstrating an uncertainty of at least one in $y \rho X$, so that y is not necessarily $i \theta X$. □

On the lower bound of (n-1) comparisons

LEMMA 1. *A selection algorithm must make exactly $n - 1$ key comparisons to select $i \,\theta\, X$, where $|X| = n$.*

Definition 1. The *key comparison* for an element $x \in X$, $x \neq i \,\theta\, X$, is defined to be the first comparison $x : y$ such that

$$y = i \,\theta\, X \text{ or } x < y < i \,\theta\, X \text{ or } i \,\theta\, X < y < x. \qquad (35)$$

(we use notation x:y to denote a comparison between elements x and y)

Proof

Firstly, we should mention that to determine which comparison is the key comparison for an element x, we must have already made all the comparisons and iθX must have already been selected.

Assume that there is an element x!=iθX that doesn't have a key comparison. This means its uncomparable with iθX, a contradiction to Theorem 1.

On the lower bound of (n-1) comparisons

LEMMA 2. *A selection algorithm must make exactly $n - 1$ joining comparisons to select $i\ \theta\ X$, where $|X| = n$.*

We will start by giving some definitions:

Definition 1

A fragment of a partial ordering (X,≤) is a maximal connected component of the partial ordering, that is, a maximal subset S of X such that the Hasse diagram of "≤" restricted to S is a connected graph.

Any partial ordering can be uniquely described up to isomorphism as the union of distinct fragments. A selection algorithm thus begins with a partial ordering consisting of n fragments of size 1. To illustrate, let $\mathfrak{F}_k$ be the set of all fragments having at most k elements:

$\mathfrak{F}_1 = \{ \bullet \}$,

$\mathfrak{F}_2 = \{\bullet,$  $\}$,

$\mathfrak{F}_3 = \{\bullet,$  $\}$, and so on.

Definition 2

A joining comparison is any comparison between elements belonging to distinct fragments

On the lower bound of (n-1) comparisons

LEMMA 2. *A selection algorithm must make exactly $n - 1$ joining comparisons to select $i \,\theta\, X$, where $|X| = n$.*

PROOF. As long as more than one fragment exists, there must be some element incomparable with $i \,\theta\, X$, since elements in distinct fragments are incomparable. The lemma then follows from Theorem 1.

ΕΥΧΑΡΙΣΤΩ ΓΙΑ ΤΗΝ ΠΡΟΣΟΧΗ ΣΑΣ