Lecture 10b: Quick Selection

Law of Least Action

Wholeness Statement

Randomized algorithms are efficient with high probability, that is, their worst case behavior is highly unlikely. Quick Select is an example. *Science of Consciousness:* The field of pure consciousness is always efficient and follows the law of least action.

Another Algorithm Design Pattern

- Prune and Search
  - AKA Decrease and Conquer
  - Examples:
    - binary search (earlier from LookupTable)
    - Downheap (earlier from QuickSort)
    - quick select
- Randomized Algorithms
  - Quick Sort, Skip Lists (Dictionary/Map), and Quick Select

Selection

Prune-and-Search
or
Decrease-and-Conquer

The Selection Problem

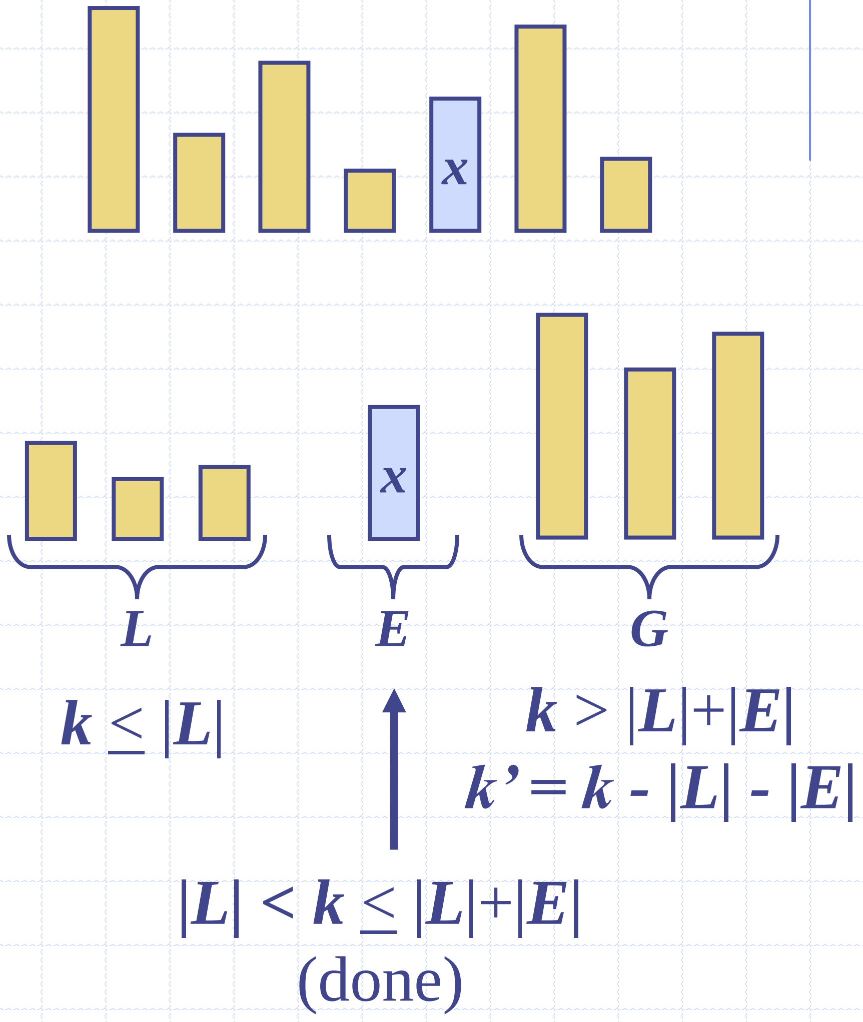
- Given an integer k and n elements $x_1, x_2, \ldots, x_n$, taken from a total order, find the k-th smallest element in this set.
- Of course, we can sort the set in O(n log n) time and then index the k-th element.

k=3 $\boxed{7\ 4\ 9\ \underline{6}\ 2 \rightarrow 2\ 4\ \underline{6}\ 7\ 9}$

- Can we solve the selection problem faster?

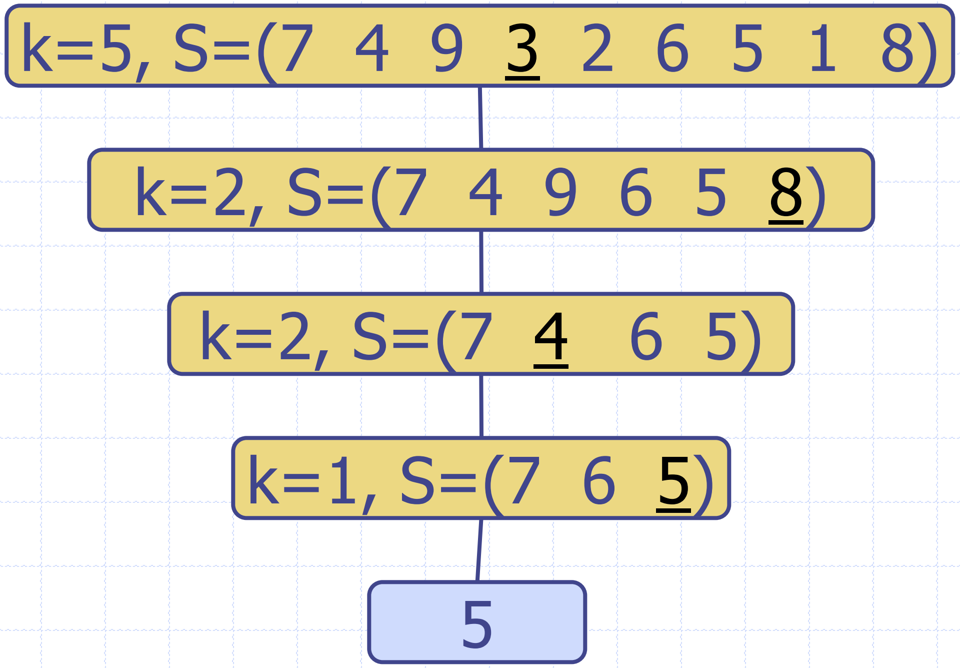
Quick-Select (§ 4.7)

- Quick-select is a **randomized** selection algorithm based on the prune-and-search paradigm:
  - Prune: pick a random element x (called pivot) and partition S into
    - L elements less than x
    - E elements equal x
    - G elements greater than x
  - Search: depending on k, either answer is in E, or we need to recurse in either L or G



Quick-Select Visualization

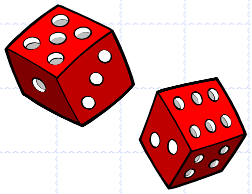
- An execution of quick-select can be visualized by a recursion path
  - Each node represents a recursive call of quick-select, and stores k and the remaining sequence



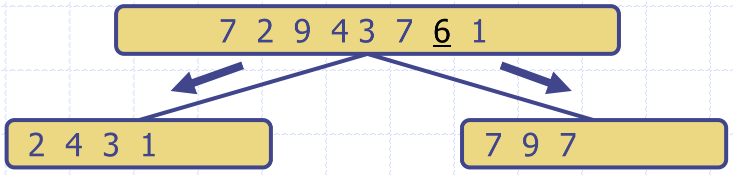
Quick Select

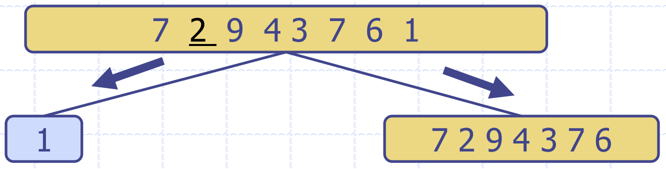
```
Algorithm QuickSelect(S, lo, hi , k )
    Input Unsorted Sequence S and k
    Output the k-th smallest element in S

    p ← inPlacePartition(S, lo, hi)
    j ← p – lo + 1
    if  j = k  then
            return S.elemAtRank(p)
    else if   j > k  then
            return QuickSelect(S, lo, p-1, k)  // T(3n/4) for good pivot
    else
            return QuickSelect(S, p+1, hi, k-j)  // T(3n/4) for good pivot
```

Expected Running Time



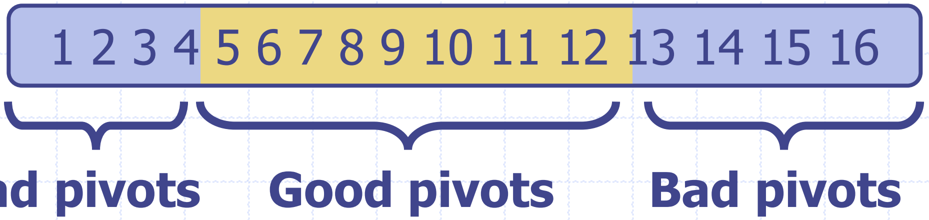
- Consider a recursive call of quick-select on a sequence of size s
  - **Good call:** the sizes of L and G are each less than $3s/4$
  - **Bad call:** one of L and G has size greater than $3s/4$





Good call **Bad call**

- A call is good with probability $1/2$
  - 1/2 of the possible pivots cause good calls:



Recurence Equations

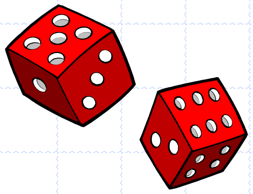
$T(n) = a\,T(n/b) + f(n)$

a=number of recursive calls made every time
1/b=fraction of input size that you recurse on
f(n) is the running time of everything except the calls

<u>QuickSelect algorithm</u>
$a=1$, $b=4/3$, $f(n)=cn$ (eliminating 1/4 is for good pivot)
$T(n)= T(3n/4) + 2cn$ (the 2 is because of bad pivots)

Master Theorem Case 3 says $T(n)=\Theta(f(n))=\Theta(n)$

Expected Running Time, Part 2



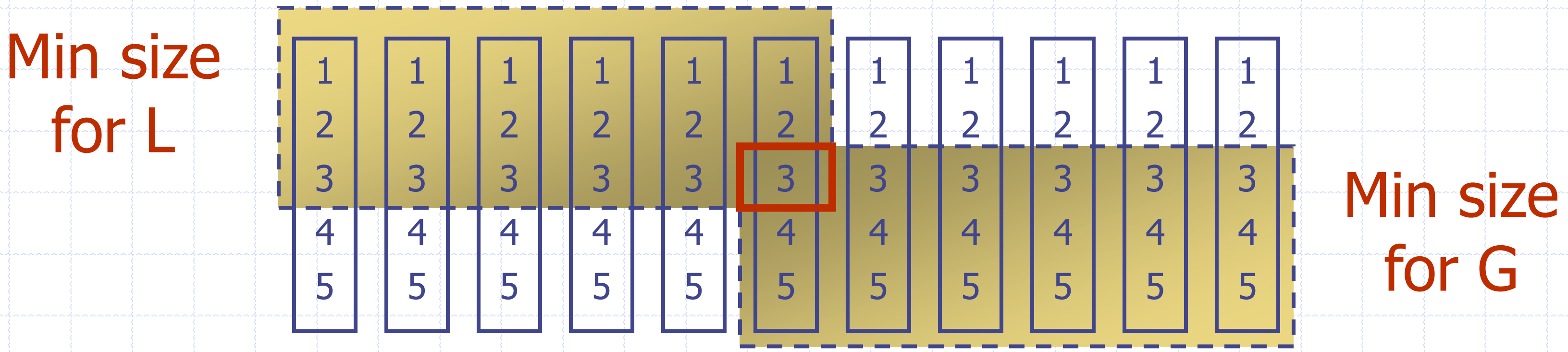
- Probabilistic Fact #1: The expected number of coin tosses required in order to get one head is two
- Probabilistic Fact #2: Expectation is a linear function:
  - $E(X + Y) = E(X) + E(Y)$
  - $E(cX) = cE(X)$
- Let T(n) denote the expected running time of quick-select.
- By Fact #2,
  - $T(n) \leq T(3n/4) + (\text{expected \# of calls before a good call}) * cn$
- By Fact #1,
  - $T(n) \leq T(3n/4) + 2cn$
- So T(n) is O(?).

Conclusion

- We can solve the selection problem in O(n) expected time

Deterministic Selection

- We can do selection in O(n) worst-case time.
- Main idea: recursively use the selection algorithm itself to find a good pivot for quick-select:
  - Divide S into n/5 sets of 5 each
  - Find a median in each set
  - Recursively find the median of the "baby" medians.



- See Exercise C-4.24 for details of analysis.

Main Point

1. Prune-and-Search algorithms reduce the search space by some fraction at each step, then the smaller problem is recursively solved.

   Science of Consciousness: The problem of world peace can be reduced to the smaller problem of peace and happiness of the individual. The problem can be further reduced to the much smaller problem of forming a small group (square root of 1%) practicing the TM and TM-Sidhi program together. Practicing together in groups is Maharishi’s approach to making the world better.

Connecting the Parts of Knowledge with the Wholeness of Knowledge

1. Selection can be done by sorting the input, then retrieving the element with the k-th smallest key (O(n log n)).

2. Applying the Prune-and-Search algorithm design strategy allows the Selection Problem to be solved in expected O(n)-time.

3. **Transcendental Consciousness** is the silent, unbounded home of all the laws of nature.
4. **Impulses within Transcendental Consciousness**: The dynamic natural laws within this unbounded field follow the law of least action that governs the activities of the universe.
5. **Wholeness moving within itself:** In Unity Consciousness, one experiences the laws of nature as waves of one’s own unbounded pure consciousness.