CSE 421
Introduction to Algorithms

Lecture 10: Divide and Conquer
Median, Quicksort
Today

Divide and conquer examples

• Simple, randomized median algorithm
  • Expected $O(n)$ time

• Surprising deterministic median algorithm
  • Worst case $O(n)$ time

• Expected time analysis for randomized QuickSort
  • Expected $O(n \log n)$ time
Order problems: Find the $k^{th}$ smallest

Runtime measures
  • # of machine instructions
  • # of comparisons

• 1$^{st}$ Smallest = Minimum
  • $O(n)$ time
  • $n - 1$ comparisons

• 2$^{nd}$ Smallest
  • Still $O(n)$ time and comparisons...
Median and Selection

**Median:** \( k \text{th smallest for } k = n/2 \)

- Easily computed in \( O(n \log n) \) time with sorting.

**Q:** How can Median be solved in \( O(n) \) time?

**A:** Use divide and conquer ...

- But Median for a smaller set isn’t a natural subproblem for Median.
- **Idea:** Generalize Median so natural subproblems are of the same type.

**Selection:**

**Given:** A (multi-)set \( S \) of \( n \) numbers, and an integer \( k \).

**Find:** The \( k \text{th smallest number in } S \).
Linear Time Divide and Conquer for Selection

General idea:

- Use a linear amount of work to reduce* Selection for a set of size $n$ to Selection for a set that is a constant factor smaller than $n$.

Recurrence

- $T(n) = T(n/b) + O(n)$ for some $b > 1$.

Apply the Master Theorem for $a = 1$, $k = 1$, and $b > 1$

- Since $a^k = 1 < b$ solution is $O(n)$.

*The value of $k$ will also change to some $k'$ for the recursive call.
General Recursive Selection

\[ \text{Select}(k, S) \]

Choose element \(x\) from \(S\) “pivot”

\[ S_L \leftarrow \{ y \in S \mid y < x \} \]
\[ S_E \leftarrow \{ y \in S \mid y = x \} \]
\[ S_G \leftarrow \{ y \in S \mid y > x \} \]

if \( |S_L| \geq k \)

return \( \text{Select}(k, S_L) \)

else if \( |S_L| + |S_E| \geq k \)

return \(x\)

else

return \( \text{Select}(k - |S_L| - |S_E|, S_G) \)
Implementing: “Choose element $x$ ...”

**Select**($k, S$)
- Choose element $x$ from $S$
- $S_L \leftarrow \{ y \in S \mid y < x \}$
- $S_E \leftarrow \{ y \in S \mid y = x \}$
- $S_G \leftarrow \{ y \in S \mid y > x \}$
- if $|S_L| \geq k$
  - return **Select**($k, S_L$)
- else if $|S_L| + |S_E| \geq k$
  - return $x$
- else
  - return **Select**($k - |S_L| - |S_E|, S_G$)

Want to choose an $x$ so that $\max(|S_L|, |S_G|)$ is as small as possible. That is, want $x$ near the middle.

Two algorithms:
- **QuickSelect**
  - Choose $x$ at random
  - Good average case performance
- **BFPRT Algorithm**
  - Choose $x$ by a complicated, but linear time method guaranteeing good split
  - Good worst case performance
QuickSelect: Random Choice of Pivot

**QuickSelect:**

- Run **Select** always choosing the pivot element $x$ uniformly at random from among the elements of $S$.

**Theorem:** QuickSelect has expected runtime $O(n)$.

**Proof:** Let $T(n)$ be the expected runtime of QuickSelect on worst-case input sets $S$ of size $n$ and integer $k$.

(The only randomness in the expectation is in the random choices of the algorithm.)
QuickSelect: Random Choice of Pivot

Consider a call to \( \text{Select}(k, S) \) and sorted order of elements in \( S \)

Elements of \( S \) listed in sorted order

With probability \( \geq 1/2 \) pivot \( x \) is good

- For any good pivot the recursive call has subproblem size \( \leq 3n/4 \)
- After 2 calls QuickSelect has expected problem size \( \leq 3n/4 \)

So \( T(n) = T(n/b) + O(n) \) for \( b = 4/3 > 1 \) \( \Rightarrow \) Expected \( O(n) \) time
Blum-Floyd-Pratt-Rivest-Tarjan Algorithm

QuickSelect requires randomness to find a good pivot and is only good on the average.

The BFPRT Algorithm always finds a good pivot that will guarantee to leave a sub-problem of size \( \leq \frac{3n}{4} \). Here is how it works...

- Split \( S \) into \( \frac{n}{5} \) sets of size 5.
- Sort each set of size 5 and choose the median of that set as its representative.
- Compute the median of those \( \frac{n}{5} \) representatives. Another recursion!
- Let the pivot \( x \) be that median.

Why does it work...?
BFPRT, Step 1: Construct sets of size 5, sort each set

**Input:**
13, 15, 32, 14, 95, 5, 16, 45, 86, 65, 62, 41, 81, 52, 32, 12, 73, 25, 81, 47, 8, 69, 9, 7, 81, 18, 25, 42, 91, 64, 98, 96, 91, 6, 51, 21, 12, 36, 11, 11, 9, 5, 17, 77

<table>
<thead>
<tr>
<th>Group:</th>
<th>13</th>
<th>5</th>
<th>62</th>
<th>32</th>
<th>47</th>
<th>81</th>
<th>64</th>
<th>51</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>16</td>
<td>41</td>
<td>12</td>
<td>8</td>
<td>18</td>
<td>98</td>
<td>21</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>45</td>
<td>81</td>
<td>73</td>
<td>69</td>
<td>25</td>
<td>96</td>
<td>12</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>86</td>
<td>52</td>
<td>25</td>
<td>9</td>
<td>42</td>
<td>91</td>
<td>36</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>95</td>
<td>65</td>
<td>32</td>
<td>81</td>
<td>7</td>
<td>91</td>
<td>6</td>
<td>11</td>
<td>77</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sort each group:</th>
<th>95</th>
<th>86</th>
<th>81</th>
<th>81</th>
<th>69</th>
<th>91</th>
<th>98</th>
<th>51</th>
<th>77</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>65</td>
<td>62</td>
<td>73</td>
<td>47</td>
<td>81</td>
<td>96</td>
<td>36</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>45</td>
<td>52</td>
<td>32</td>
<td>9</td>
<td>42</td>
<td>91</td>
<td>21</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>16</td>
<td>41</td>
<td>25</td>
<td>8</td>
<td>25</td>
<td>64</td>
<td>12</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>5</td>
<td>32</td>
<td>12</td>
<td>7</td>
<td>18</td>
<td>6</td>
<td>11</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

$O(n)$
### BFPRT, Step 2: Find median of column medians

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>95</td>
<td>86</td>
<td>81</td>
<td>81</td>
<td>69</td>
<td>91</td>
<td>98</td>
<td>51</td>
<td>77</td>
</tr>
<tr>
<td>32</td>
<td>65</td>
<td>62</td>
<td>73</td>
<td>47</td>
<td>81</td>
<td>96</td>
<td>36</td>
<td>17</td>
</tr>
</tbody>
</table>

**Column medians:**

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>45</td>
<td>52</td>
<td>32</td>
<td>9</td>
<td>42</td>
<td>91</td>
<td>21</td>
<td>11</td>
</tr>
<tr>
<td>14</td>
<td>16</td>
<td>41</td>
<td>25</td>
<td>8</td>
<td>25</td>
<td>64</td>
<td>12</td>
<td>9</td>
</tr>
<tr>
<td>13</td>
<td>5</td>
<td>32</td>
<td>12</td>
<td>7</td>
<td>18</td>
<td>6</td>
<td>11</td>
<td>5</td>
</tr>
</tbody>
</table>

*Imagining rearranging columns by column median*

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>95</td>
<td>86</td>
<td>81</td>
<td>81</td>
<td>69</td>
<td>91</td>
<td>98</td>
<td>51</td>
<td>77</td>
</tr>
<tr>
<td>32</td>
<td>65</td>
<td>62</td>
<td>73</td>
<td>47</td>
<td>81</td>
<td>96</td>
<td>36</td>
<td>17</td>
</tr>
</tbody>
</table>

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>45</td>
<td>52</td>
<td>32</td>
<td>9</td>
<td>42</td>
<td>91</td>
<td>21</td>
</tr>
<tr>
<td>14</td>
<td>16</td>
<td>41</td>
<td>25</td>
<td>8</td>
<td>25</td>
<td>64</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>5</td>
<td>32</td>
<td>12</td>
<td>7</td>
<td>18</td>
<td>6</td>
<td>11</td>
</tr>
</tbody>
</table>

\(T(n/5)\)
BFPRT, Step 2: Find median of column medians

<table>
<thead>
<tr>
<th></th>
<th>95</th>
<th>86</th>
<th>81</th>
<th>81</th>
<th>69</th>
<th>91</th>
<th>98</th>
<th>51</th>
<th>77</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>65</td>
<td>62</td>
<td>73</td>
<td>47</td>
<td>81</td>
<td>96</td>
<td>36</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>45</td>
<td>52</td>
<td>32</td>
<td>9</td>
<td>42</td>
<td>91</td>
<td>21</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>16</td>
<td>41</td>
<td>25</td>
<td>8</td>
<td>25</td>
<td>64</td>
<td>12</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>5</td>
<td>32</td>
<td>12</td>
<td>7</td>
<td>18</td>
<td>6</td>
<td>11</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

Column medians: $T(n/5)$

Imagining rearranging columns by column medians
### BFPRT, Step 2: Find median of column medians

<table>
<thead>
<tr>
<th>Column medians:</th>
<th>T(n/5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>95 86 81 81 69 91 98 51 77</td>
<td></td>
</tr>
<tr>
<td>32 65 62 73 47 81 96 36 17</td>
<td></td>
</tr>
<tr>
<td>15 45 52 32 9 42 91 21 11</td>
<td></td>
</tr>
<tr>
<td>14 16 41 25 8 25 64 12 9</td>
<td></td>
</tr>
<tr>
<td>13 5 32 12 7 18 6 11 5</td>
<td></td>
</tr>
</tbody>
</table>

Choose \( x \) to be that median of medians

<table>
<thead>
<tr>
<th>Not in ( S_G )</th>
<th>Size ( \geq n/4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>95 51 77 69 81</td>
<td>91 98 86 81</td>
</tr>
<tr>
<td>32 36 17 47</td>
<td>73 81 96 65</td>
</tr>
<tr>
<td>15 21 11</td>
<td>9</td>
</tr>
<tr>
<td>14 12 9</td>
<td>8</td>
</tr>
<tr>
<td>13 11 5</td>
<td>7</td>
</tr>
<tr>
<td>42 91 45 52</td>
<td>25 64 16 41</td>
</tr>
<tr>
<td>18 6</td>
<td>5</td>
</tr>
</tbody>
</table>
BFPRRT, Step 2: Find median of column medians

Column medians:

<table>
<thead>
<tr>
<th>95</th>
<th>86</th>
<th>81</th>
<th>81</th>
<th>69</th>
<th>91</th>
<th>98</th>
<th>51</th>
<th>77</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>65</td>
<td>62</td>
<td>73</td>
<td>47</td>
<td>81</td>
<td>96</td>
<td>36</td>
<td>17</td>
</tr>
<tr>
<td>15</td>
<td>45</td>
<td>52</td>
<td>32</td>
<td>9</td>
<td>42</td>
<td>91</td>
<td>21</td>
<td>11</td>
</tr>
<tr>
<td>14</td>
<td>16</td>
<td>41</td>
<td>25</td>
<td>8</td>
<td>25</td>
<td>64</td>
<td>12</td>
<td>9</td>
</tr>
<tr>
<td>13</td>
<td>5</td>
<td>32</td>
<td>12</td>
<td>7</td>
<td>18</td>
<td>6</td>
<td>11</td>
<td>5</td>
</tr>
</tbody>
</table>

Choose $x$ to be that median of medians

<table>
<thead>
<tr>
<th>95</th>
<th>51</th>
<th>77</th>
<th>69</th>
<th>81</th>
<th>91</th>
<th>98</th>
<th>86</th>
<th>81</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>36</td>
<td>17</td>
<td>47</td>
<td>73</td>
<td>81</td>
<td>96</td>
<td>65</td>
<td>62</td>
</tr>
<tr>
<td>15</td>
<td>21</td>
<td>11</td>
<td>9</td>
<td>32</td>
<td>42</td>
<td>91</td>
<td>45</td>
<td>52</td>
</tr>
<tr>
<td>14</td>
<td>12</td>
<td>9</td>
<td>8</td>
<td>25</td>
<td>25</td>
<td>64</td>
<td>16</td>
<td>41</td>
</tr>
<tr>
<td>13</td>
<td>11</td>
<td>5</td>
<td>7</td>
<td>12</td>
<td>18</td>
<td>6</td>
<td>5</td>
<td>32</td>
</tr>
</tbody>
</table>

Not in $S_L$

Size $\geq n/4$

$|S_L|, |S_G| \leq \frac{3n}{4}$
BPFRT Recurrence

Choose partitioning element \( x \)
- \( T(n/5) + O(n) \)

Partitioning based on \( x \)
- \( O(n) \)

Cost of recursive subproblem
- \( T(3n/4) \)

Recurrence
- \( T(n) = T(3n/4) + T(n/5) + O(n) \)

Why is the solution \( O(n) \)?
Solution to \( T(n) = T(3n/4) + T(n/5) + cn \) is \( O(n) \)

Key property of recurrence:

- \( 3/4 + 1/5 < 1 \)
- Sum is \( 19/20 \)

Cost at top level is \( cn \); so at other levels, linear in the sum of problem sizes:

- Sum of problem sizes decreases by \( 19/20 \) factor per level of recursion
- Total cost is geometric series with ratio \( < 1 \) and largest term \( cn \)
- Solution is \( O(n) \).
QuickSort

QuickSort($S$)

if $|S| \leq 1$ return $S$

Choose element $x$ from $S$  "pivot"

$S_L \leftarrow \{ y \in S \mid y < x \}$

$S_E \leftarrow \{ y \in S \mid y = x \}$

$S_G \leftarrow \{ y \in S \mid y > x \}$

return [QuickSort($S_L$), $S_E$, QuickSort($S_G$)]
QuickSort

Pivot selection

• Choose the median
  • $T(n) = 2 \ T(n/2) + O(n) \quad O(n \log n)$

• Choose arbitrary element
  • Worst case – $O(n^2)$
    • Element might be smallest, so one subproblem has size $n - 1$
  • Average case – $O(n \log n)$ similar to QuickSelect analysis

• Choose random pivot
  • Expected time – $O(n \log n)$

We’ll give an analysis for this bound ...
Expected Runtime for QuickSort: “Global analysis”

Runtime is proportional to # of comparisons
  • Count comparisons for simplicity

Master theorem kind of analysis won’t work ...

Instead, use a clever global analysis:
  • Number elements $a_1, a_2, \ldots, a_n$ based on final sorted order
  • Let $p_{ij} =$ Probability that QuickSort compares $a_i$ and $a_j$

Expected number of comparisons:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} p_{ij}$$
Expected Runtime for QuickSort: “Global analysis”

Lemma: For $i < j$ we have $p_{ij} \leq \frac{2}{j-i+1}$.

Proof: If $a_i$ and $a_j$ are compared then it must be during the call when they end up in different subproblems

• Before that, they aren’t compared to each other
• After they aren’t compared to each other

During this call they are only compared if one of them is the pivot

All elements between $a_i$ and $a_j$ are also in the call:

• $\Rightarrow$ set has size at least $j - i + 1$ in this call
• Probability one of the 2 is chosen as pivot is $\leq \frac{2}{j - i + 1}$. ■
Expected Runtime for QuickSort: “Global analysis”

Lemma: For $i < j$ we have $p_{ij} \leq \frac{2}{j-i+1}$.

Expected number of comparisons:

$$
\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} p_{ij} \leq \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} \\
= \sum_{i=1}^{n-1} \sum_{k=1}^{n-i+1} \frac{2}{k+1} < 2 \sum_{i=1}^{n-1} \sum_{k=1}^{n} \frac{1}{k} < 2n H_n \\
= 2n \ln n + O(n) \leq 1.387n \log_2 n
$$

Harmonic series sum:

$$H_n = \sum_{k=1}^{n} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \ldots + \frac{1}{n}$$

Fact: $H_n = \ln n + O(1)$
QuickSort in Practice (Nonrandom)

Separating out set $S_E$ of elements equal to the pivot is important
- Use 4-finger algorithm instead of 2-finger algorithm for partitioning
  - Collect equal elements at each end and swap to middle at end of
    partitioning (saves a lot on size of recursive set sizes)
- If $n$ is very small use InsertionSort instead (also good if set is nearly sorted)
- Small $n$
  - choose middle element of subarray as pivot
- Medium $n$
  - choose median of 3 elements as pivot
- Large $n$
  - consider 9 elements in 3 groups of 3; choose median of medians as pivot