Randomized Quick Sort Algorithm

Lecture Note – 1

Prepared By: Muhammed Miah
Randomized Quick Sort

- In traditional Quick Sort, we always pick the first element as the pivot for partitioning.
- The worst case runtime is $O(n^2)$ while the expected runtime is $O(n \log n)$ over the set of all input.
- Therefore, some input are born to have long runtime, e.g., an inversely sorted list.
Randomized Quick Sort

- In randomized Quick Sort, we pick randomly an element as the pivot for partitioning.
- The expected runtime of any input is $O(n \log n)$.
- Randomized Quick Sort is a probabilistic algorithm.
Randomized Quick Sort (S)

- **Input:** a set of numbers $S$, $S = \{S_1, S_2, S_3, ..., S_n\}$
- **If** $|S| > 1$ Then,
  1. Select an element $X$ from $S$ at random
  2. Partition $S$ into 3 parts: L X R
  3. Random Quick Sort (L);
  4. Random Quick Sort (R);
Randomized Quick Sort

Example:

- $S = \{2, 1, 3, 5, 4\}$

- Sorted: 1 2 3 4 5

Diagram:

- 1st pivot: 2
- 2nd pivot: 3
- 3rd pivot: 5
- 4th pivot: 4
- 5th pivot: 3

01/16/2008
Randomized Quick Sort

- Properties:
  - Running time is unpredictable
  - It has randomization built into it
  - Nothing to do with input order
  - Worst unequal division – when one side has zero number and other side has all numbers
  - Worst case running time $n^2$
## Analysis of “Expected” Running Time of Randomized QS

- Assume, $S = \{1, 2, 3, 4, \ldots, n\}$. Let us consider a Boolean variable, $X_{ij}$, $1 \leq i, j \leq n$, $j > i$
- $X_{ij} = 1$ if it is compared with $j$
- $X_{ij} = 0$ otherwise
- # of variables we can create $\approx n^2$

<table>
<thead>
<tr>
<th></th>
<th>$X_{12}$</th>
<th>$X_{13}$</th>
<th>\ldots</th>
<th>$X_{n-1}$</th>
<th>SUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 if $X_1$ and $X_2$ are compared, 0 otherwise</td>
<td>1</td>
<td>1</td>
<td>\ldots</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>millions of trees are generated</td>
<td>1</td>
<td>1</td>
<td>\ldots</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

AVG. $= \frac{\text{SUM}}{n}$

AVG($X_{12}$) $= \frac{\text{SUM}({X_{12}})}{n}$

AVG($X_{13}$) $= \frac{\text{SUM}({X_{13}})}{n}$

AVG($X_{ij}$) $= \frac{\text{SUM}({X_{ij}})}{n}$

SUM($X_{12}$, $X_{13}$, ..., $X_{n-1}$) $= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}$

Millions of trees are generated
Analysis of “Expected” Running Time of Randomized QS

- $X_{ij}$ is a random variable

\[
E\left[ \sum_{i=1}^{n-1} \sum_{j>i}^{n} X_{ij} \right] = \sum_{i=1}^{n-1} \sum_{j>i}^{n} E[X_{ij}]
\]

<table>
<thead>
<tr>
<th>Horizontal definition</th>
<th>Vertical definition</th>
</tr>
</thead>
</table>

Linearity of expectation,

\[
E[X + Y] = E[X] + E[Y]
\]

$E[X_{ij}] = \text{how often } i \text{ and } j \text{ compared}$
Analysis of “Expected” Running Time of Randomized QS

- Level wise left to right (LR) order:
  - Consider set of numbers \{1, 2, ..., 10\}
  - Assume we got the tree:
  - Level wise order:
    6 4 8 3 5 9 1 10 2

If Min(LR[i], LR[j]) > Min(LR of i+1, i+2, ..., j-1), then
i is NOT compared to j
otherwise it is compared

01/16/2008
Analysis of “Expected” Running Time of Randomized QS

\[ P(X_{ij} = 1) = \frac{2}{j-i+1} \]

\[ E[X_{ij}] = \frac{2}{j-i+1} \]

\[
\sum_{i=1}^{n-1} \sum_{j>i}^{n} E[X_{ij}] = \sum_{i=1}^{n} \sum_{j>i}^{n} \frac{2}{j-i+1}
\]

\[
\approx \sum_{i=1}^{n} \sum_{j=i}^{n} \frac{2}{j-i} \approx \sum_{i=1}^{n} \sum_{k=1}^{n-i} \frac{2}{k} \quad \text{let, } j-i = k
\]

\[
\leq \sum_{i=1}^{n} \sum_{k=1}^{n} \frac{2}{k}
\]

01/16/2008
Analysis of “Expected” Running Time of Randomized QS

\[ 2 \sum_{i=1}^{n} \sum_{k=1}^{n} \frac{1}{k} = 2n \left( \sum_{k=1}^{n} \frac{1}{k} \right) \]

\[ \sum_{k=1}^{n} \frac{1}{k} = \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n} \]

(Harmonic Series)

\[ 2n \left( \sum_{k=1}^{n} \frac{1}{k} \right) = 2n \log n \approx O(n \log n) \]

So, “Expected” running time of randomized quick sort = \( O(n \log n) \)