CSE548 & AMS542: Analysis of Algorithms, Fall 2019 **Lecturer:** Rezaul A. Chowdhury
Topic: High Probability Bound for RandQS and Random Skip Lists **Date:** 20^{th} Nov 2019
Scribe: Abhishek Kumar

22.1 High Probability Bound : RandQS

From the last lecture we saw the expected running time of randomized Quick Sort to be $\mathcal{O}(nlogn)$. In this lecture we will see that with high probability (w.h.p) in n, the running time does not exceed its expected running time by more than a constant factor. In other words w.h.p the running time of randomized Quick Sort is $\mathcal{O}(nlogn)$.

Fix an arbitrary element z in the input. After partition, z will end up in one of the two partitions and this continues till the point when z becomes a pivot where it stops, as shown in the figure below.

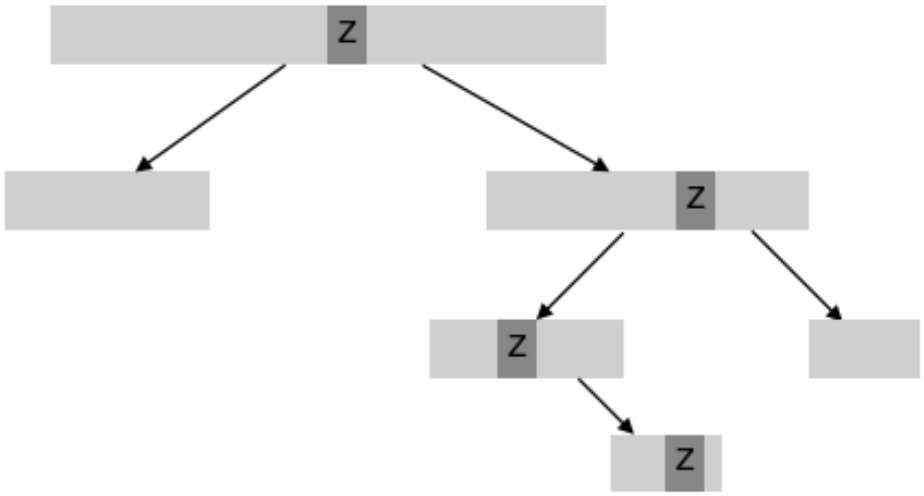

Figure 22.1.1: z propagation

The idea is to show w.h.p in n, z will stop within $32 \ln n$ partitions. In randomized QuickSort, 50:50 or 25:75 split is not guaranteed at every partition. The best possible scenario is 50:50 split which terminates after k partition steps from the equation $n(\frac{1}{2})^k = 1$, therefore k = $\log_2 n$ partitions. Similarly for a 25:75 split ratio, the equation becomes $n(\frac{3}{4})^k = 1$, solving for k, we get k = $\log_{\frac{4}{3}} n$ partitions, which is also logarithmic in n.

If we guarantee that the partition has at most $\frac{3n}{4}$ elements at every step, then we observe the partition levels to still remain logarithmic in n. This guarantee is not possible but we will show with high probability in n and $32 \ln n$ or $c \ln n$ partioning steps, at least $\ln n$ of them are balanced or good partitions - meaning they contain atmost $\frac{3n}{4}$ elements.

z will undergo many partitioning steps some of which will be good and others bad. A partitioning step is

good(or balanced) if size of the larger partition is at most $\frac{3n}{4}$, bad(or unbalanced) otherwise.

To ensure neither partition is greater than $\frac{3n}{4}$, the pivot p must satisfy, $\frac{n}{4} \leq rank(p) \leq \frac{3n}{4}$.

Then the probability of choosing such a pivot becomes $(\frac{3n}{4} - \frac{n}{4})/n$ which equals $\frac{1}{2}$.

Therefore if we choose a pivot uniformly at random, the probability of ending up with a good partition is $\frac{1}{2}$.

Lemma 22.1.1 *If $c \ln n$ be the total number of partitions for c > 0 then w.h.p in n, atleast $\frac{c}{4} \ln n$ of them are good(or balanced) partitions.*

Proof: Let there be n elements in the input and $c \ln n$ be the total number of partitions possible.

After k partitioning steps, we are left with at most $(\frac{3}{4})^k n$ elements.

Since the input size is n, after $\frac{c}{4} \ln n$ balanced partitions,

z will end up in a partition of size

$$\leq \frac{3}{4}^{(\frac{c}{4} \ln n)} = \frac{n}{n^{\frac{c}{4} \ln \frac{4}{3}}} \text{ which is } \leq 1, \text{ for c } \geq 14.$$

In other words, z will end up in the final sorted position after $\frac{c}{4} \ln n$ balanced partitions.

Now we need to show that with high probability in n, there are $\frac{c}{4} \ln n$ balanced partitions.

For $1 \leq i \leq c \ln n$, let

$$Z_i = \begin{cases} 1 & \text{if the partition at recursion level i is balanced;} \\ 0 & \text{otherwise.} \end{cases} \tag{22.1.1}$$

Total number of balanced partitioning steps is

$$Z = \sum_{i=1}^{c \ln n} Z_i$$

Since $P_r[Z_i = 1] = \frac{1}{2}, E[Z_i] = \frac{1}{2}$.

Therefore

$$\mu = E[Z] = \sum_{i=1}^{c \ln n} E[Z_i] = \frac{c}{2} \ln n$$

From Chernoff bounds we have,

$$0 < \delta < 1, P_r(Z \leq (1 - \delta)\mu) \leq e^{\frac{-\mu \delta^2}{2}}.$$

Applying the above chernoff bound with $\delta = \frac{1}{2}$ and $\mu = \frac{c}{2} \ln n$,

$$\implies P_r\Big(Z \leq \Big(1 - \frac{1}{2}\Big)\frac{c}{2} \ln n\Big) \leq e^{-(\frac{1}{2})^2 \frac{c}{4} \ln n}$$

$$\implies P_r(Z \leq \frac{c}{4}\ln n) \leq \frac{1}{e^{(\frac{c}{16}\ln n)}} \leq \frac{1}{n^{\frac{c}{16}}} \leq \frac{1}{n^2}, \text{ with c } = 32.$$

$\therefore$ The probability that balanced partitioning steps is less than $\frac{c}{4}\ln n$ is $\leq \frac{1}{n^2}$ which is very low .

In other words, the probability that z fails to reach its final sorted position after $32\ln n$ partitions is $\left(\frac{1}{n^2}\right)$. ■

We now need to show that this remains consistent with all n items.

The probability that at least one of the n items is not in final sorted position is

$$\leq \sum_{i=1}^{n} P_r(\text{item}_i \text{ is not in final sorted position}) \leq \sum_{i=1}^{n} \frac{1}{n^2} \leq \frac{1}{n}$$

$\therefore$ The probability that all n input elements reach their final sorted positions after $32\ln n$ levels of recursion is

$$\geq \left(1 - \frac{1}{n}\right) \leftarrow \text{ high probability}$$

Observe that the total amount of work done at each level of recursion is bounded by the cost of partition to be $\mathcal{O}(n)$ since we lose the pivot at every partition.

So, the total work done in $32\ln n$ levels of recursion is $\mathcal{O}((32\ln n)n) = \mathcal{O}(n\ln n)$.

Therefore, we prove w.h.p in n, RandQS terminates in $\mathcal{O}(n\ln n)$ time.

22.2 Random Skip Lists

Searching in a traditional sorted Linked List is $\mathcal{O}(n)$.

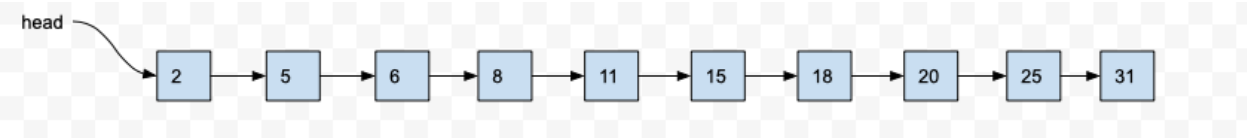

Figure 22.2.2: Sorted Linked List

22.2.1 2-level Linked List

To reduce the search time asymptotically, we can promote certain items in the linked list to a higher level and link the levels. Create segments of length $\sqrt{n}$ and promote first item of each segment to the next higher level. The number of items promoted in the below example is $\sqrt{n}$, where each promoted item has link to an item in the just lower level and a link to the next promoted item if it exists. Therefore we observe the search cost in a 2-level linked list to be $2\sqrt{n}$.

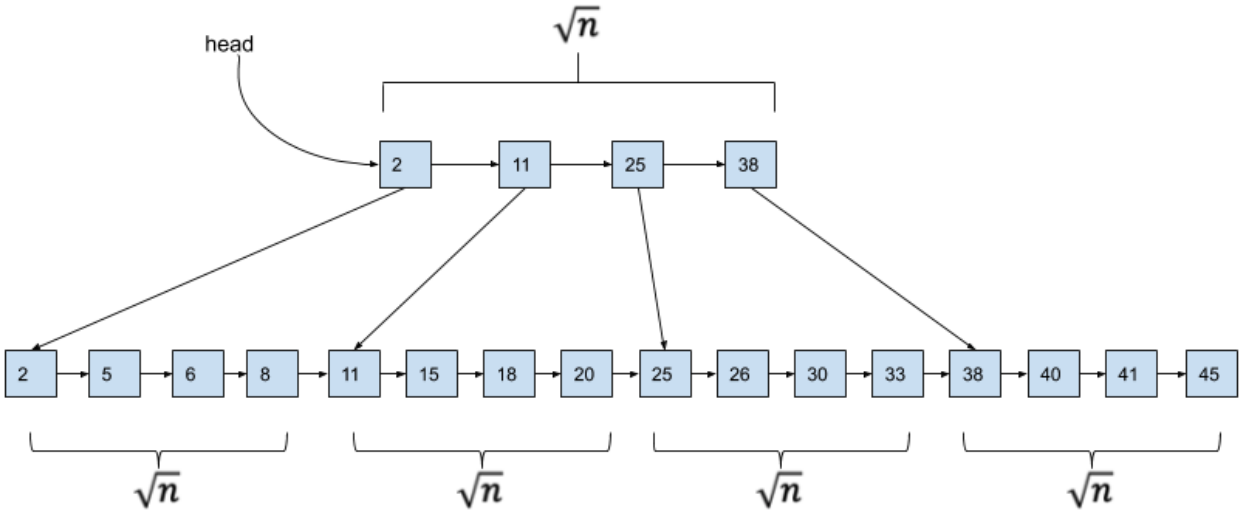

Figure 22.2.3: 2 Level Sorted Linked List

22.2.2 3-level Linked List

Similarly to reduce search time further, a 3-level linked list can be created from $\sqrt[3]{n}$ sized segments by promoting $n^{\frac{2}{3}}$ items to the 2^{nd} level and then $n^{\frac{1}{3}}$ items again to the 3^{rd} level. We observe that the search cost reduces to $3\sqrt[3]{n}$.

Question: For the 3-level linked list, why did we use $\sqrt[3]{n}$ as the segment size and not $\sqrt{n}$?

Answer: Asymptotically, the time complexity for search would remain $\sqrt{n}$ and not improve since we would have to traverse $\sqrt{n}$ items in the bottom most level in the worst case.

In general, for a k-level Linked List the search time is $k\sqrt[k]{n}$. What is the limit, how far can we go ? As we decrease the segment size we see the number of levels increase, therefore at some point the two terms will balance each other.

Let us assume k = $\log_2 n$.

$$\therefore \text{Search cost} = (\log_2 n)n^{(\frac{1}{\log_2 n})} = (\log_2 n)n^{\log_n 2} = 2\log_2 n$$

22.2.3 (logn)-level Linked List

Search is most efficient in this kind of Linked List but inserts and deletes become too complicated to perform.

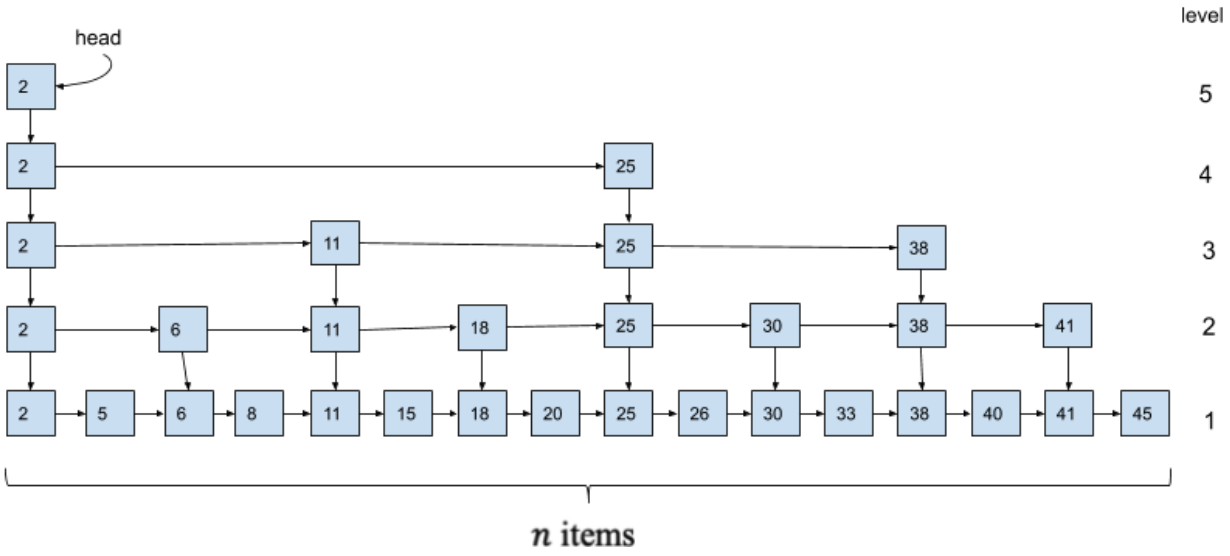

Figure 22.2.4: Deterministic (logn)-level Linked List

22.2.4 Random Skip list

Insert and delete operations can be done with ease when the data structure involves randomization as we will see. The promotion of items at every level does not remain deterministic anymore but becomes probabilistic, similar to the tossing of a fair coin at every item - heads meaning promotion and no promotion otherwise. Flipping of a coin at every item is an independent trial and does not depend on other coin tosses. We also observe that it is possible for an item to reach the highest level through promotion but the probability of such an event is low. An item with $-\infty$ at the starting is present for convenience at every level.

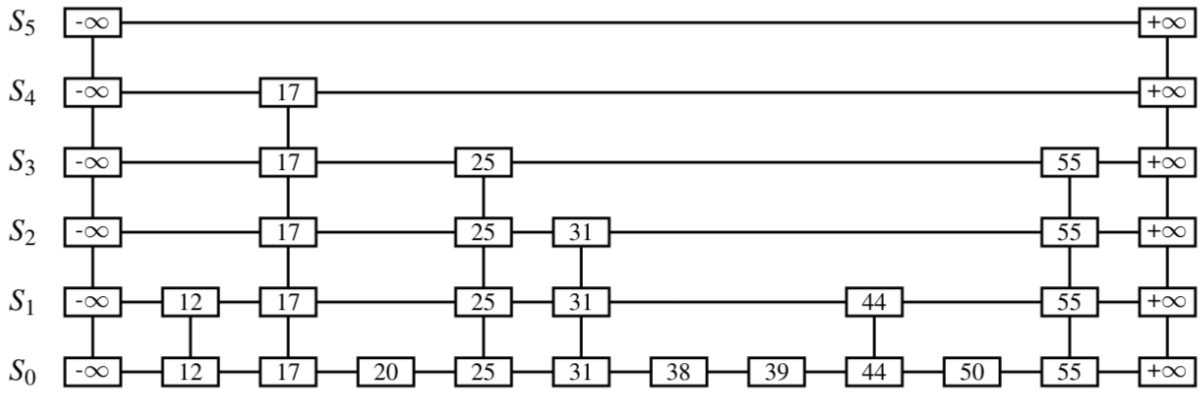

Figure 22.2.5: Random Skip List

Deletion becomes easy - find that item and remove the whole column and fix the pointers.

Insertion also becomes easy - find the position where the item needs to go, insert it and then by flipping coins copy it over to higher levels and fix the pointers.

In both operations, searching that item is necessary and cost of search determines the time complexity for insert and delete. We'll try to figure out the height of the random skip list.

Let height of List be $h[L]$.

$$P_r[h[L]] \leq k = \prod_{x \in L} P_r[h[x]] \leq k]$$

$$x \in L, h[x] > k = \sum_{i=k+1}^{\infty} \frac{1}{2^i} = \frac{1}{2^k}$$

$$\implies h[x] \leq k = 1 - \frac{1}{2^k}$$

$$\therefore P_r[h[L] \leq k] = \prod_{x \in L} P_r[h[x]] \leq k] = \left(1 - \frac{1}{2^k}\right)^n$$

Substituting $k = c \log_2 n$ and $c > 2$,

$$\left(1 - \frac{1}{2^k}\right)^n = \left(1 - \frac{1}{n^c}\right)^n \geq \left(1 - \frac{n}{n^c}\right) \geq \left(1 - \frac{1}{n^{c-1}}\right)$$

$$\therefore P_r[h[L] \leq c \log_2 n] \geq \left(1 - \frac{1}{n^{c-1}}\right) \text{ which is very high (highly probable in n)}$$

W.H.P, height of random skip list is $\Theta(\log_2 n)$.

On close observation, it can be seen that the search cost is not only dependent on the number of downward moves(height of random skip list) but also on the number of sidewise moves. We will use the nature of coin tosses to prove w.h.p the expected number of sidewise movements.

Lets say we flip $4c \log_2 n$ coins.

If the i^{th} coin toss turns out to be heads, $Z_i = 1$.

$$\text{The total number of heads } Z = \sum_{i=1}^{4c \log_2 n} Z_i$$

$$\mu = E[Z] = \sum_{i=1}^{4c \log_2 n} E[Z_i] = \sum_{i=1}^{4c \log_2 n} \frac{1}{2} = 2c \log_2 n$$

Applying Chernoff bound,

$$\delta = \frac{1}{2} \text{ and } \mu = 2c \log_2 n$$

$$P_r[Z \leq c \log_2 n] \leq e^{\left(-\frac{\left(\frac{1}{2}\right)^2 2c \log_2 n}{2}\right)} = \frac{1}{n^{\frac{c}{4}}}$$

Therefore w.h.p in n, for $4c \log_2 n$ coin tosses, at least $c \log_2 n$ of them are heads.

Since the search path corresponds to sequence of coin tosses, w.h.p in n - no search path can be longer than $4c \log_2 n$. $c \log_2 n$ of them correspond to heads(upwards movement) and the rest $3c \log_2 n$ correspond to tails(sideways movement).

Hence we show that search cost is $\mathcal{O}(\log_2 n)$.

22.3 References

- https://cs.stackexchange.com/questions/101337/distribution-of-pointer-keys-in-a-skip-list-node