Optimal Bounds for Open Addressing Without Reordering

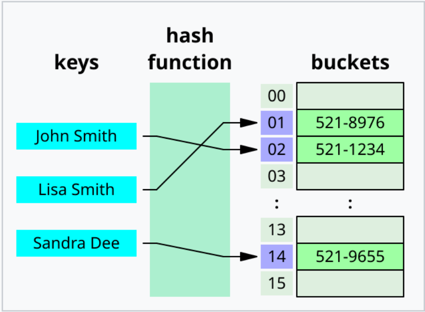
Martin Farach-Colton , Andrew Krapivin , William Kuszmaul

Presented by - Amit Roy

March 27, 2025

Hash Table

- Support dictionary operations - INSERT, SEARCH and DELETE
- Uses a hash function $h : [u] \rightarrow [m]$ to index keys



Collision Resolutions

- Chaining - Each slot in the table is a pointer to a linked list which stores the keys
- Open Addressing - All elements occupy the hash table itself

Chaining vs Open-Addressing

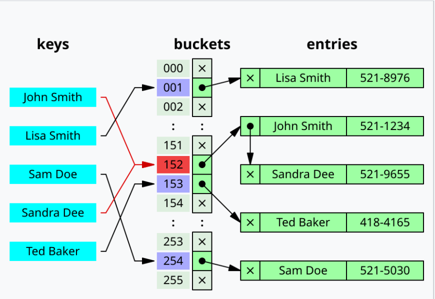


Figure: Chaining

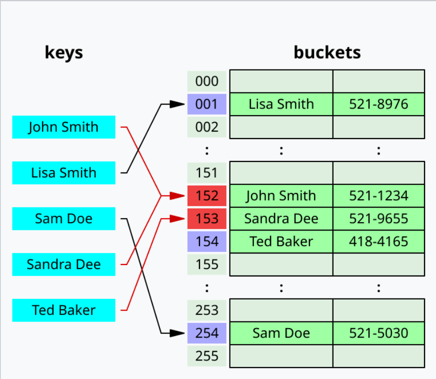


Figure: Open Addressing

1

Definitions

Probe Complexity

- The number of probes that an algorithm has to make to insert/search the key is called the probe complexity of the key.
- For example, for a key k, if an algorithm probes $h_1(k), h_2(k), \ldots, h_t(k)$ to find an empty slot to insert the key, then the probe complexity of k is t.

Uniform Probing

For a given key k, the probe sequence - $h_1(k), h_2(k), \ldots, h_t(k)$ is a random permutation of $\{1, 2, \ldots, n\}$

Greedy and Non-greedy Open-Addressing

- **Greedy :** Any algorithm in which each element uses the first unoccupied position in its probe sequence.
- **Non-greedy :** May probe further before inserting the element in the hash table

Results

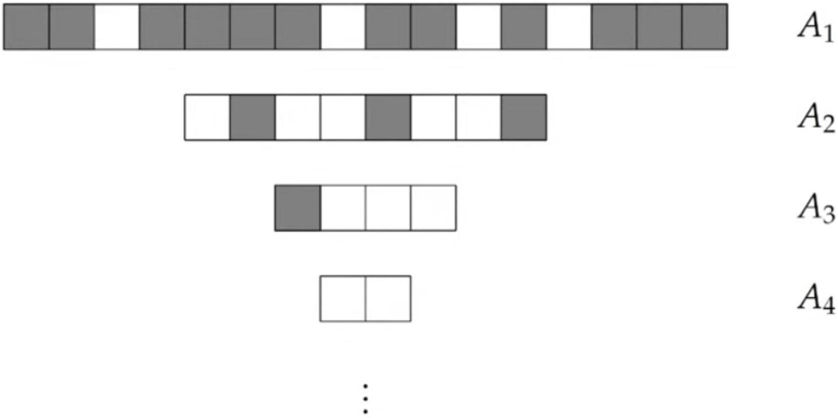
1. Greedy
   - Worst-case expected probe complexity $\mathcal{O}(\log^2 \delta^{-1})$
   - High-probability worst-case probe complexity $\mathcal{O}(\log^2 \delta^{-1} + \log\log n)$
     - ⋆ Matching lower bound
2. Non-Greedy
   - Amortized probe complexity $\mathcal{O}(1)$
   - Worst-case expected probe complexity $\mathcal{O}(\log \delta^{-1})$
     - ⋆ Matching lower bound

Theorem - Greedy Open-Addressing

Let $n \in \mathbb{N}$ and $\delta \in (0,1)$ be parameters such that $\delta > \mathcal{O}(1/n^{o(1)})$. There exists a **greedy** open-addressing strategy that supports $n - \lfloor \delta n \rfloor$ insertions that has

- worst-case expected probe complexity (and insertion time) - $\mathcal{O}(\log^2 \delta^{-1})$
- worst-case probe complexity over all insertions - $\mathcal{O}(\log^2 \delta^{-1} + \log\log n)$, with prob $1 - 1/poly(n)$,
- amortized expected probe complexity - $\mathcal{O}(\log \delta^{-1})$

Funnel Hashing



Funnel Hashing

Algorithm 1: Insert key k into the hash table

```
for i = 1 to α do
    if Insertion_Attempt(i, k) is successful then
        return;
    end
end
Insert into special array A_{α+1}
```

Funnel Hashing

Algorithm 2: Insertion Attempt of key k in A_i

Hash k to obtain a subarray index $j \in \left[\frac{|A_i|}{\beta}\right]$;

for *each slot in* $A_{i,j}$ **do**

  if *slot is empty* **then**

    Insert key and return success;

  end

end

Return fail;

Algorithm for special array $A_{\alpha+1}$

1. Split $A_{\alpha+1}$ into two subarrays B and C of equal size.
2. First, try to insert in B. Upon failure insert into C (insertion to C is guaranteed to succeed with high probability)
3. B is implemented as a uniform probing table, and we give up searching through B after $\log \log n$ attempts.
4. C is implemented as a two-choice table with buckets of size $2 \log \log n$.

Proof

Lemma 1

For a given $i \in \alpha$, we have with probability $1 - \frac{1}{n^{\omega(1)}}$ that, after $2|A_i|$ insertion attempts have been made in A_i, fewer than $\frac{\delta}{64}|A_i|$ slots in A_i remain unfilled.

Lemma 2

The number of keys inserted into $A_{\alpha+1}$ is fewer than $\frac{\delta}{8}n$, with probability $1 - \frac{1}{n^{\omega(1)}}$.

Proof

Lemma: Power of two choices

If m balls are placed into n bins by choosing two bins uniformly at random for each ball and placing the ball into the emptier of the two bins, then the maximum load of any bin is $m/n + \log\log n + \mathcal{O}(1)$ with high probability in n.

Probe Complexity of $A_{\alpha+1}$

Complexity of inserting into B -

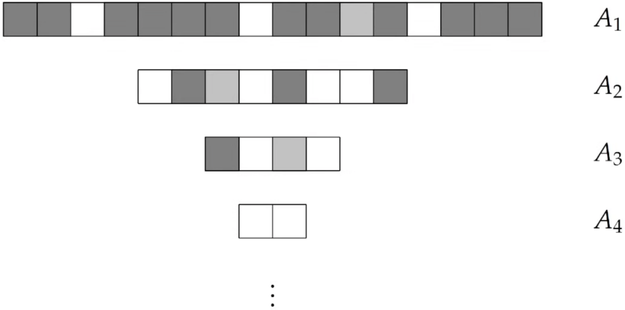
1. B has size $A_{\alpha+1}/2 \geq \delta n/4$, so load factor never exceeds $1/2$.
2. Each insertion makes $\log\log n$, each of which has success probability of $1/2$.
3. Thus, expected number of probles is $\mathcal{O}(1)$
4. Probability that insertion fails after all attempts is $1/2^{\log\log n} \leq 1/\log n$.

Probe Complexity of $A_{\alpha+1}$

Complexity of inserting into C -

1. Recall, C is implemented as a two choice table with buckets of size $2 \log \log n$
2. From Lemma we have that, with high probability, no bucket in C overflows.
3. Expected time of each insertion in C is at most $o(1)$.

Analysis



- $k = O(\log \delta^{-1})$ levels
- Level cutoff $c = O(\log \delta^{-1})$
- Worst case (expected) probe complexity: $ck = O(\log^2 \delta^{-1})$

Proof

- Total α arrays , and cutoff probes β in each A_i
- Probe complexity of each insertion - $\beta\alpha + f(A_{\alpha+1})$
- Assume $\delta \leq \frac{1}{8}$. Let $\alpha = \lceil 4 \log \delta^{-1} + 10 \rceil$ and $\beta = \lceil 2 \log \delta^{-1} \rceil$
- Probe Complexity - $\mathcal{O}(\log^2 \delta^{-1}) + f(A_{\alpha+1})$
- Hence, $\mathcal{O}(\log^2 \delta^{-1})$ in worst-case expected probe complexity and a high-probability worst-case probe complexity of $\mathcal{O}(\log^2 \delta^{-1} + \log \log n)$.

Other Results

1. **Elastic Hashing**

Theorem - Non-greedy Open-Addressing

Let $n \in \mathbb{N}$ and $\delta \in (0,1)$ be parameters such that $\delta > \mathcal{O}(1/n)$. There exists an open-addressing hash table that supports $n - \lfloor \delta n \rfloor$ insertions in an array of size n, that does not reorder items after they are inserted, and that offers -

- amortized expected probe complexity $O(1)$
- worst-case expected probe complexity $\mathcal{O}(log \delta^{-1})$, and
- worst-case expected insertion time $\mathcal{O}(log \delta^{-1})$.

Other Results

2. **Lower Bounds**

Theorem - Lower Bound for Greedy Algorithms

Let $n \in \mathbb{N}$ and $\delta \in (0, 1)$ be parameters such that δ is an inverse power of two. Consider any greedy open-addressed hash table with capacity n. If $(1 - \delta)n$ elements are inserted into the hash table, then the final insertion must take expected time $\Omega(\log^2 \delta^{-1})$.

More Proofs

Lemma 2

The number of keys inserted into $A_{\alpha+1}$ is fewer than $\frac{\delta}{8}n$, with probability $1 - \frac{1}{n^{\omega(1)}}$.

- From **Lemma 1**, every *fully-explored* A_i is at least $(1 - \delta/64)$ full, where *fully-explored* means at least $2|A_i|$ insertion attempts made to A_i.
- Let $\lambda \in [\alpha]$ be largest index s.t. A_λ receives fewer than $2|A_\lambda|$ insertion attempts.
- **Case 1:** $\lambda \leq \alpha - 10$
  - For $i > \lambda$, A_i contains at least $|A_i|(1 - \delta/64)$ keys.
  - Total keys in $i \geq \lambda$: $(1 - \delta/64) \sum_{i=\lambda+1}^{\alpha} |A_i| \geq 2.5(1 - \delta/64)|A_\lambda|$
  - This contradicts that A_λ received at most $2|A_\lambda|$ insertion attempts.

Lemma 2 Proof Contd

- **Case 2 :** $\alpha - 10 < \lambda \leq \alpha$
  - Fewer than $A_{\alpha-10} < n\delta/8$ keys are attempted to be inserted in A_i with $i \geq \lambda$. Hence, we are good.
- **Case 3:** $\lambda = null$
  - Each A_i has at most $\delta|A_i|/64$ empty slots.
  - Total empty slots at the end of insertion : $|A_{\alpha+1}| + \sum_1^\alpha \frac{\delta|A_i|}{64} < n\delta$
  - This contradicts that after $n(1-\delta)$ insertions, there are at least $n\delta$ slots empty.

Proof of Lemma 1

Lemma 1

For a given $i \in \alpha$, we have with probability $1 - \frac{1}{n^{\omega(1)}}$ that, after $2|A_i|$ insertion attempts have been made in A_i, fewer than $\frac{\delta}{64}|A_i|$ slots in A_i remain unfilled.

Thoughts and Questions ?

Thanks