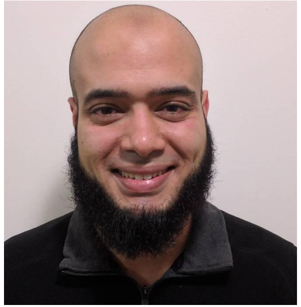
Data Structures

Hash Table

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



Searching for a string

- Assume we have to implement a data structure to support the following:
    - Contains 1-to-1 mapping
    - Insertion of a string (in O(L), where L = letters)
    - Checks if a string exists
- How to solve?
    - We can use a linked list? Insert is O(L) and Search is O(NL)
        - O(L) comes from copying a string or comparing 2 strings.
    - We can use an AVL tree? Insert and search is O(L logN)
- Can we check it much more quickly?

Using hashing

- Assume we have a hash function that creates no collision for N = 10 millions
  - Assume O(L) for hashing a string of L letters
- How can we use it for very fast insert/check?
  - Create a boolean array of 10 million
  - For a string S, compute hash_code in O(L)
  - Mark in O(1) for insertion in the array: array[hash_code] = 1
  - Check in O(1) if it exists in the array or not
- Great that we mark/check in O(1)!
  - However, there's generally no such hash function
  - There are collisions!
  - Can we handle the collisions so that we are hopefully closer to O(1)?

Without collisions (wrong assumption)

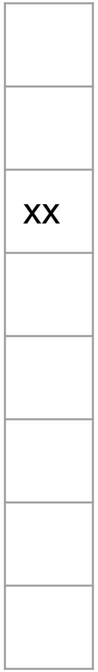
String	Hash Code
mostafa	2
ali	5
wow	0
ibrahim	3

1
1
1
1

- In the (big) array, mark positions 0, 2, 3, 5 for our strings
- Perfect!

With collisions (realistic)

String	Hash Code
mostafa	2
ali	5
wow	**2 (collision)**
ibrahim	3



- Now index 2 is used by 2 strings
- We can’t simply marks this in the array
- There's an easy trick for this. Think about it for 5 minutes

Collision Resolution

- There are 2 ways to handle the collision problem
- **Chaining** (*aka Open Addressing, closed hashing*)
  - Use a **separate** data structure to handle collisions
  - E.g. An array where each cell is a linked list of items with the same keys
- **Probing** (*aka Closed Addressing, open hashing*)
  - We use a single array, but with some strategy to systematically use/change the array

“Acquire knowledge and impart it to the people.”

“Seek knowledge from the Cradle to the Grave.”