## #26: Hashing: Collision Handling

October 25, 2019 · *G Carl Evans*

Every hash table contains three pieces:

1. A **hash function**, **f(k)**: keyspace → integer
2. **An array**.
3. **A collision handling strategy.**

---

### Collision Handling Strategy #1: Separate Chaining

Example: **S = { 16, 8, 4, 13, 29, 11, 22 }, |S| = n**
**h(k) = k % 7,** **|Array| = N**

| | |
|---|---|
| [0] | |
| [1] | |
| [2] | |
| [3] | |
| [4] | |
| [5] | |
| [6] | |
| [7] | |

**Load Factor:**

**Running time of Separate Chaining:**

| | **Worst Case** | **SUHA** |
|---|---|---|
| **Insert** | | |
| **Remove/Find** | | |

### Collision Handling Strategy #2: Probe-based Hashing

Example: **S = { 16, 8, 4, 13, 29, 11, 22 }, |S| = n**
**h(k) = k % 7,** **|Array| = N**

| | |
|---|---|
| [0] | |
| [1] | |
| [2] | |
| [3] | |
| [4] | |
| [5] | |
| [6] | |
| [7] | |

**Linear Probing:**

Try h(k) = (k + 0) % 7, if full...
Try h(k) = (k + 1) % 7, if full...
Try h(k) = (k + 2) % 7, if full...
...

**What problem occurs?**

---

### Double Hashing:

Example: **S = { 16, 8, 4, 13, 29, 11, 22 }, |S| = n**
**$h_1$(k) = k % 7, $h_2$(k) = 5 – (k % 5)**, **|Array| = N**

| | |
|---|---|
| [0] | |
| [1] | |
| [2] | |
| [3] | |
| [4] | |
| [5] | |
| [6] | |
| [7] | |

**Double Hashing:**

Try h(k) = (k + + 0*$h_2$(k)) % 7, if full...
Try h(k) = (k + + 1*$h_2$(k)) % 7, if full...
Try h(k) = (k + + 2*$h_2$(k)) % 7, if full...
...

h(k, i) = ($h_1$(k) + i*$h_2$(k)) % 7

**Running Time:**

<u>Linear Probing:</u>
- Successful: $\frac{1}{2}(1 + 1/(1-\alpha))$
- Unsuccessful: $\frac{1}{2}(1 + 1/(1-\alpha))^2$

<u>Double Hashing:</u>
- Successful: $1/\alpha * \ln(1/(1-\alpha))$
- Unsuccessful: $1/(1-\alpha)$

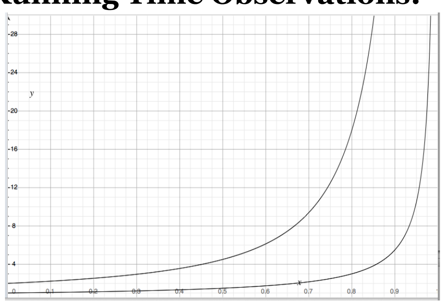
<u>Separate Chaining:</u>
- Successful: $1 + \alpha/2$
- Unsuccessful: $1 + \alpha$

**Running Time Observations:**

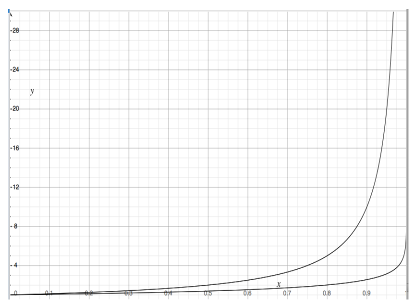
1. As $\alpha$ increases:

2. If $\alpha$ is held constant:

**Running Time Observations:**



**Linear Probing:**
Successful: $\frac{1}{2}(1 + 1/(1-\alpha))$
Unsuccessful: $\frac{1}{2}(1 + 1/(1-\alpha))^2$



**Double Hashing:**
Successful: $1/\alpha * \ln(1/(1-\alpha))$
Unsuccessful: $1/(1-\alpha)$

---

**ReHashing:**
What happens when the array fills?

Better question:

Algorithm:

**Which collision resolution strategy is better?**
- Big Records:
- Structure Speed:

**What structure do hash tables replace?**

**What constraint exists on hashing that doesn't exist with BSTs?**

**Why talk about BSTs at all?**

**Analysis of Dictionary-based Data Structures**

| | Hash Table | | AVL | List |
|---|---|---|---|---|
| | Amortized | Worst Case | | |
| **Find** | | | | |
| **Insert** | | | | |
| **Storage Space** | | | | |

---

**A Secret, Mystery Data Structure:**

**ADT:**
insert

remove

isEmpty

**CS 225 – Things To Be Doing:**

1. Programing Exam B is on-going
2. MP5 has been released; EC+7 deadline is Monday night
3. lab_btree due Sunday
4. Daily POTDs are ongoing!