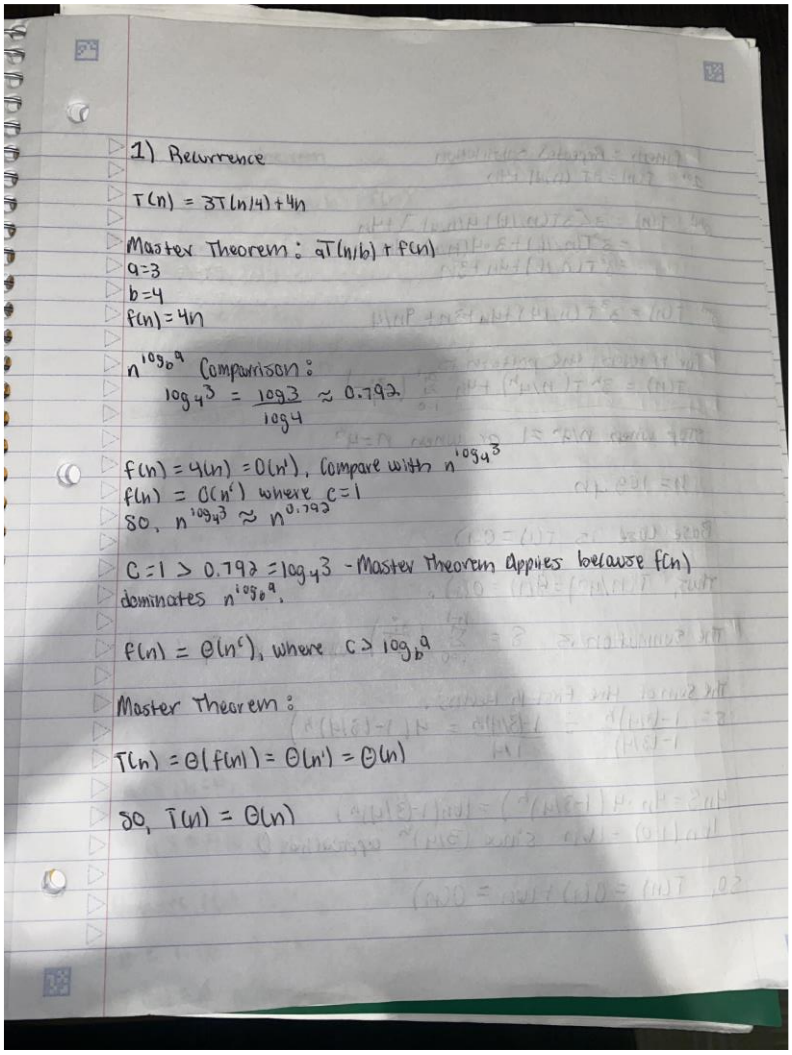
Bethaney Mallory-Smothers

Introduction to Algorithms (CSC 3130-02)

Dr Parra-Rodriguez

20 March 2025

**HashMaps**



Pattern = Repeated Substitution

1st $T(n) = 3T(n/4) + 4n$

2nd $T(n) = 3[3T(n/16) + 4(n/4)] + 4n$
$= 3^2T(n/16) + 3 \cdot 4(n/4) + 4n$
$= 3^2T(n/16) + 4n + 3n$

3rd $T(n) = 3^3T(n/64) + 4n + 3n + 9n/4$

For $n$ levels, the pattern is
$T(n) = 3^kT(n/4^k) + 4n \sum_{i=0}^{k-1} \left(\frac{3^i}{4^i}\right)$

Stop when $n/4^k = 1$ or when $n = 4^k$

$k = \log_4 n$

Base case is $T(1) = O(1)$

Thus, $T(n/4^k) = T(1) = O(1)$.

The summation is $S = \sum_{i=0}^{k-1} \left(\frac{3^i}{4^i}\right)$ where $r = \frac{3}{4}$

The sum of the first $k$ terms,
$S = \frac{1-(3/4)^k}{1-(3/4)} = \frac{1-(3/4)^k}{1/4} = 4(1-(3/4)^k)$

$4nS = 4n \cdot 4(1-(3/4)^k) = 16n(1-(3/4)^k)$
$16n(1-0) = 16n$ since $(3/4)^k$ approaches 0

So, $T(n) = O(1) + 16n = O(n)$

2) Master Theorem

$T(n) = aT(n/b) + f(n)$

a) $T(n) = 3T(n/5) + n^3$
$a=3, b=5, f(n) = n^3$

$\log_5 3 \approx 0.6826$

Compare $f(n)$ and $n^{\log_5 3}$ → $3 > 0.6826$, case 3 applies

$T(n) = \Theta(n^3)$

b) $T(n) = 4T(n/3) + Tn$
$a=4, b=3, f(n) = Tn$

$\log_3 4 \approx 1.2619$

Compare $f(n) \tilde{=} O(n^1)$ with $n^{\log_3 4}$

$1 < 1.2619$ → case 1 applies

$T(n) = \Theta(n^{\log_3 4}) = \Theta(n^{1.2619})$

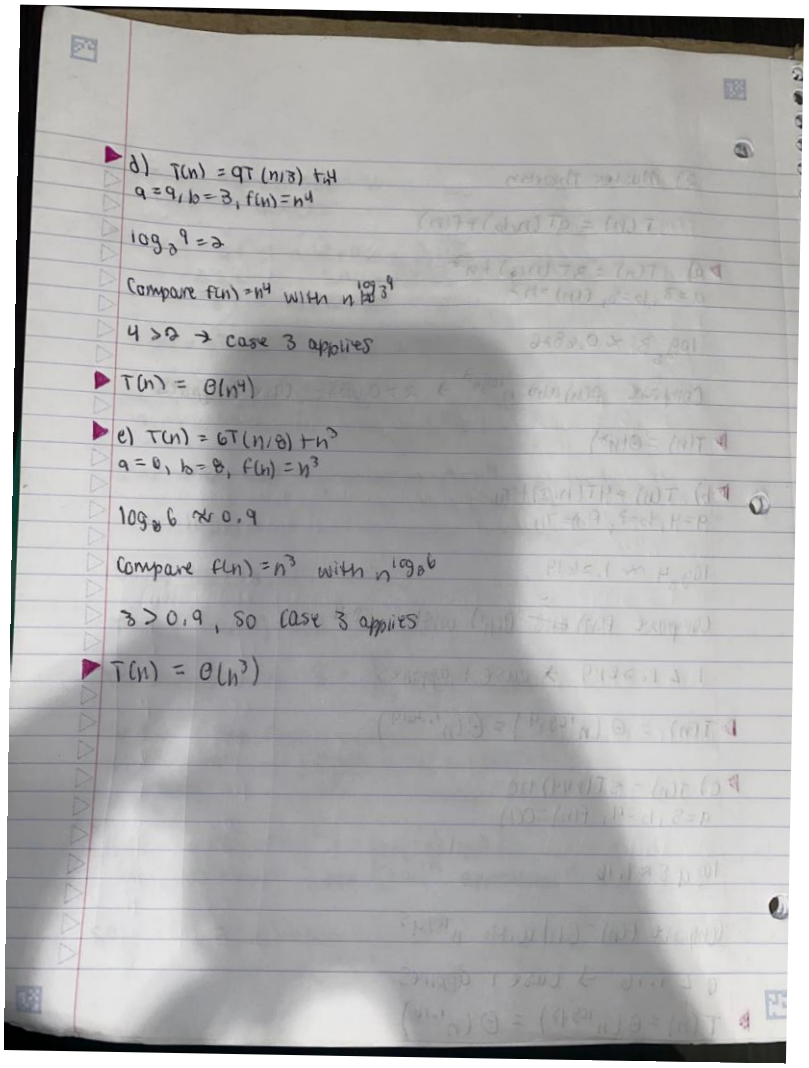
c) $T(n) = 5T(n/4) + 10$
$a=5, b=4, f(n) = O(1)$

$\log_4 5 \approx 1.16$

Compare $f(n) = O(1)$ with $n^{\log_4 5}$

$0 < 1.16$ → case 1 applies

$T(n) = \Theta(n^{\log_4 5}) = \Theta(n^{1.16})$




**3. Radix Sort**

**Sort by last letter:** CAD, DOD, DOG, FIG, CAP, CAR, JOB, COL, COX, LOL, LOW, LOX, RAT, ROW, PIG, SUN, TSL, USD, VEA, VEE, VIS, WOW

**Sort by middle letter:** CAD, CAP, CAR, COX, COL, DOD, DOG, FIG, JOB, LOL, LOW, LOX, PIG, RAT, ROW, SUN, TSL, USD, VEA, VEE, VIS, WOW

**Sort by first letter:** CAD, CAP, CAR, COX, COL, DOD, DOG, FIG, JOB, LOL, LOW, LOX, PIG, RAT, ROW, SUN, TSL, USD, VEA, VEE, VIS, WOW

**Final:** CAD, CAP, CAR, COX, COL, DOD, DOG, FIG, JOB, LOL, LOW, LOX, PIG, RAT, ROW, SUN, TSL, USD, VEA, VEE, VIS, WOW

4) Double Hashing

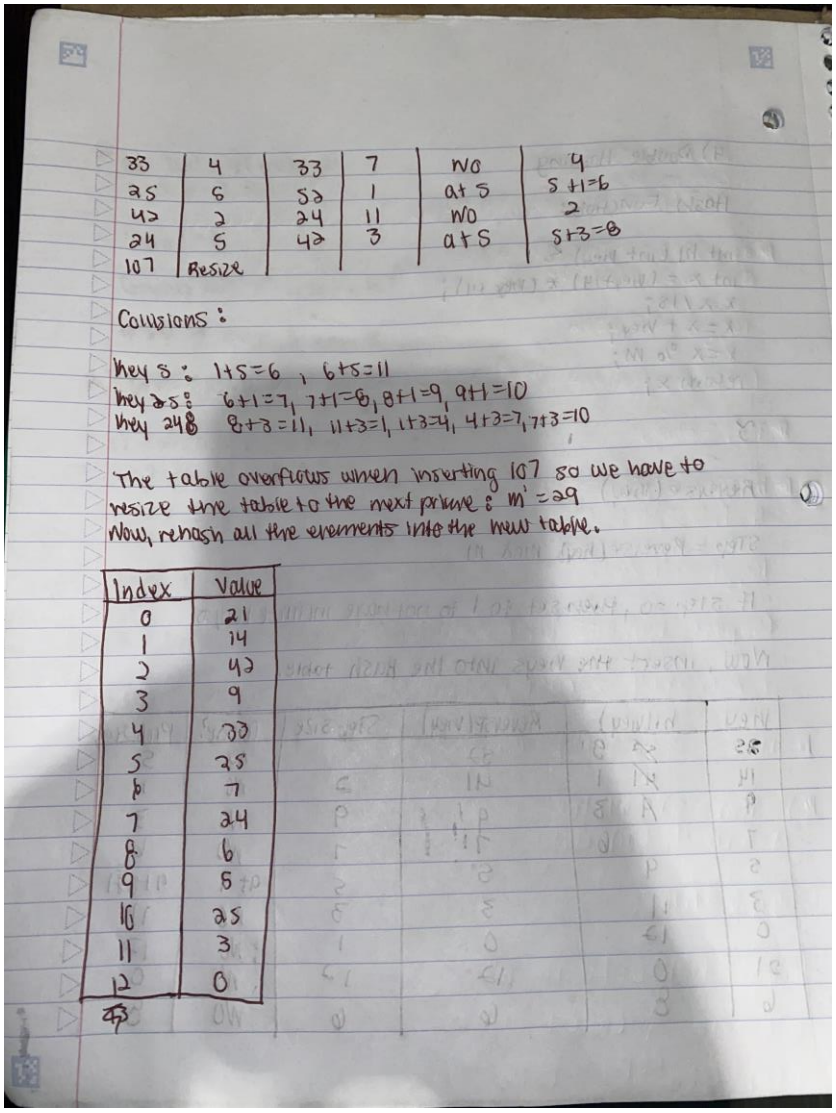
Hash Functions:

```
int h1(int key) {
int x = (key+19) * (key+41);
x = x / 15;
x = x + key;
x = x % M;
return x;
}

Reverse(key)

Step = Reverse(key) mod M
```

If step = 0, then set to 1 to not have infinite loops

Now, insert the keys into the hash table.

| key | h1(key) | Reverse(key) | Step size | Collide? | Probe |
|---|---|---|---|---|---|
| ~~25~~ | ~~8~~ 9 | 52 | | NO | 5 |
| 14 | ~~11~~ 1 | 41 | 2 | NO | 1 |
| 9 | ~~4~~ 3 | 9 | 9 | NO | 3 |
| 7 | 6 | 7 | 7 | NO | 6 |
| 5 | 9 | 5 | 5 | at 9 | 9+5=14 |
| 3 | 11 | 3 | 3 | NO | 11 |
| 0 | 12 | 0 | 1 | NO | 12 |
| 21 | 0 | 12 | 12 | NO | 0 |
| 6 | 8 | 6 | 6 | NO | 8 |



| | | | | | |
|---|---|---|---|---|---|
| 33 | 4 | 33 | 7 | NO | 4 |
| 25 | 6 | 52 | 1 | at 5 | 5+1=6 |
| 42 | 2 | 24 | 11 | NO | 2 |
| 24 | 5 | 42 | 3 | at 5 | 5+3=8 |
| 107 | Resize | | | | |

Collisions:

key 5: 1+5=6, 6+5=11
key 25: 6+1=7, 7+1=8, 8+1=9, 9+1=10
key 248: 8+3=11, 11+3=1, 1+3=4, 4+3=7, 7+3=10

The table overflows when inserting 107 so we have to resize the table to the next prime: m' = 29
Now, rehash all the elements into the new table.

| Index | Value |
|---|---|
| 0 | 21 |
| 1 | 14 |
| 2 | 42 |
| 3 | 9 |
| 4 | 33 |
| 5 | 25 |
| 6 | 7 |
| 7 | 24 |
| 8 | 6 |
| 9 | 5 |
| 10 | 25 |
| 11 | 3 |
| 12 | 0 |

43

## 7. Algorithm Analysis

### Radix Sort for Strings

**Time Complexity**: O(NM) where N is the number of strings in the array and M is the maximum string length. Each pass takes O(N), and we perform M passes.

**Space Complexity**: O(N) due to the temporary buckets used in sorting.

**Word Pattern Matching**

**Time Complexity**: O(N), where N is the length of the string. We traverse the string once and use hash maps for constant-time lookups.

**Space Complexity**: O(N), as we store mappings in two hash maps.

**Subarray Sum (EC Problem)**

**Time Complexity**: O(N), as we iterate through the array once using the sliding window approach.

**Space Complexity**: O(1), since we only use a few integer variables.