

# Hashing

---

Trideep Gogoi  
157 B

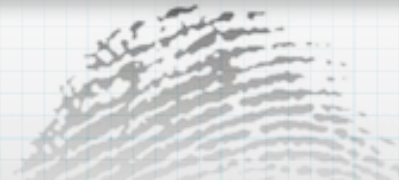
# Topics

- \* What is Hashing?
- \* Why do we need it?
- \* Properties.

# What is Hashing?

- \* Like a Digital fingerprint for data.
- \* Take data and generate a unique "ID tag" for it.
- \* Data can then be identified via ID tag

One person One  
fingerprint



# One atomic Value one "hash"

Hi My name is Trideep and I am a Software Engineer.  
I like this field because it will enable me to make a  
lot of money. Hopefully over \$100,000 a year.  
That is unless of-course my Job gets outsourced to India!



Hash Function



4356

# How to generate a hash?

$P = \text{"My name is Trideep"}$

$H = \text{Hash}(P)$

$H = \text{"2190"}$

# How to compute a Hash?

- \* No formal Definition.
- \*  $\text{Hash}(\text{"P"}) = \text{P}$  is still **valid**.

# Examples of Hashes.

- \* **mod** function
- \* Student ID is "004082901".
- \*  $004082901 \bmod 9999$ .
- \* **3309**

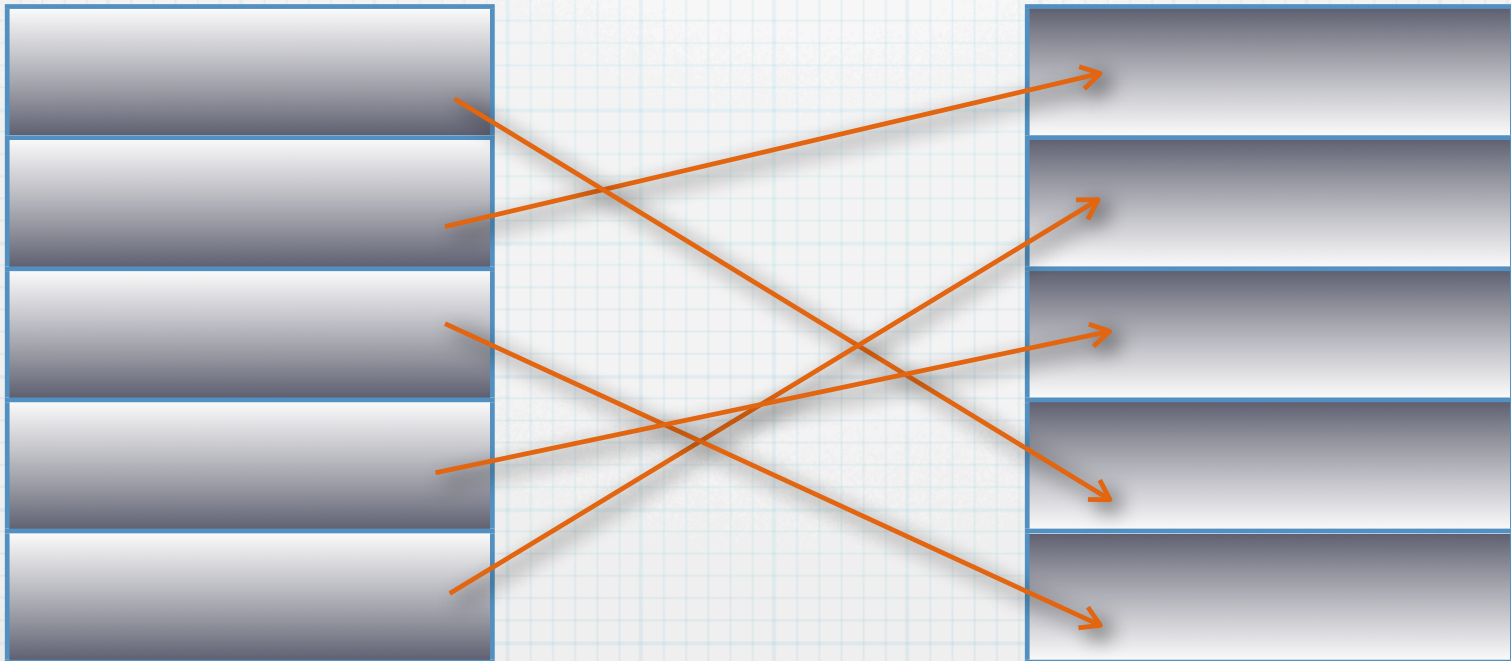
# Mod Function

- \* Gives the remainder.
- \* Pseudo random Even Distribution.

# Mod Function

- \* Range... 0 to 500
- \*  $4578 \bmod 500 = 78$
- \*  $8903 \bmod 500 = 403$
- \*  $9830 \bmod 500 = 330$

# Mod function



# More examples.

- \* Folding

- \* ALBERT

- \* 27 42 36 17 11 19

- \* 2742 + 3671 + 1119

- \* 7478 mod 101 = 52

# Midsquare

$$(453)^2$$

20529

52

# Why Hashing?

- \* What are the ways to search or reference in a database?

# Why Hashing?

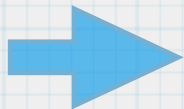
- \* Sequential Search
- \* Index Search
- \* Direct Access Hashing

# Sequential Search.

- \* Go through records one by one comparing to search value.

# Sequential Search.

Name	ID	City
Ackbar	093	LAX
Anakin	005	SFO
Vader	029	HON
Solo	208	CLT
Luke	678	SJC



# Problems

- \* Large Databases Not feasible.
- \* Databases Stored in Disks Not memory
- \* Very large databases stored over multiple disks.
- \* Even multiple locations...

# Indexed Search.

- \* Build Index of smaller values.
- \* Find value in index table.
- \* Use index to determine position in Database.
- \* Indexes are small. Can be stored in memory.
- \* For large Database same **problem** as Sequential search

# Indexed Search.

Name	RecNo
R2D2	04
Chew	03
Jabba	02
Lila	01

RECno	Name	ID	City
01	Lila	026	SFO
02	Jabba	109	LAX
03	Chew	367	NYC
04	R2D2	423	DEL



# Another Approach.

- \* Direct Access.
- \* Directly go to Record needed.
- \* **No** search or comparisons (ideally).

# Use Hashes

- \* Use Hash function to calculate Record Number/index/location.
- \* Create Hash function to map **Key** to a index.
- \* Hash (**Key**) gives index on Database.

# Hash Search.

Hash ("Jabba") = 02

RECno	Name	ID	City
01	Lila	026	SFO
02	Jabba	109	LAX
03	Chew	367	NYC
04	R2D2	423	DEL



# Use Hashes.

\* No comparisons made!

# Hash Use.

Hash (982567) = 2665

Recno	Name	ID
710	Luke	320678
7825	Jabba	897736
7536	Chewbacca	897447
2665	Darth	982567
1221	Obi	981123



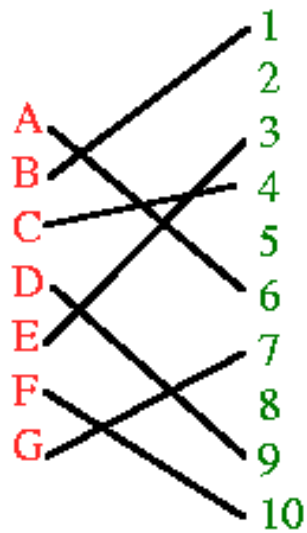
# Properties.

- \* Uniform Distribution.
- \* Reduce Collision.
- \* Collision : Two different Keys map to same.

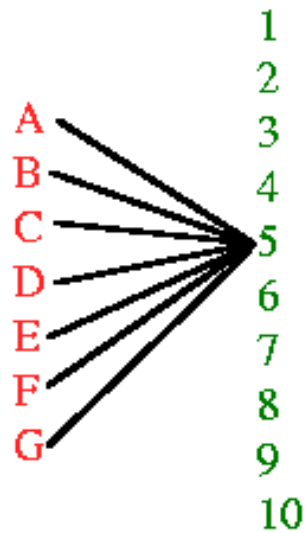
# Collision.

## Good and Bad Functions

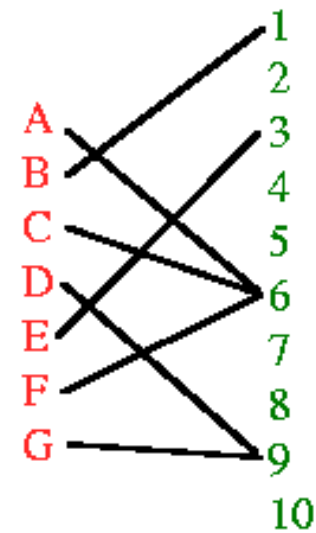
Best



Worst



Acceptable



# Questions/Doubts?

## \* References

- \* Information Security Principles and Practice by Mark Stamp
- \* Wikipedia.
- \* Professor Lee's Slides