Data Structures and Algorithms
Hashing

CS 225
G Carl Evans

April 17, 2023

Department of Computer Science
Learning Objectives

Motivate and formally define a hash table

Discuss what a ‘good’ hash function looks like

Identify the key weakness of a hash table

Introduce strategies to “correct” this weakness
Data Structure Review

I have a collection of books and I want to store them in a dictionary!

What data structures can I use here?
I have a collection of books and I want to store them in a dictionary!

<table>
<thead>
<tr>
<th>Find</th>
<th>Insert</th>
<th>Remove</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sorted Array</td>
<td>BST</td>
<td>AVL Tree</td>
</tr>
</tbody>
</table>

- **Sorted Array**
- **BST**
- **AVL Tree**
What if $O(\log n)$ isn’t good enough?
What if $O(\log n)$ isn’t good enough?
Hash Function

Maps a **keyspace**, a (mathematical) description of the keys for a set of data, to a set of integers.

![Diagram showing a hash function mapping keys to values](image)
Hash Function

A hash function *must* be:

- **Deterministic:**
- **Efficient:**
- **Defined for a certain size table:**
Hash Function

(Angreve, CS 341)
( Beckman, CS 101)
(Challon, CS 125)
(Davis, CS 105)
(Evans, CS 225)
(Fagen-Ulmschneider, CS 107)
(Gunter, CS 422)
(Herman, CS 233)
Hash Function

(Angrave, CS 341)
(Beckman, CS 101)
(Challon, CS 125)
(Davis, CS 105)
(Evans, CS 225)
(Fagen-Ulmschneider, CS 107)
(Gunter, CS 422)
(Herman, CS 233)

Hash function
(key[0] - ‘A’)

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angrave</td>
<td>341</td>
</tr>
<tr>
<td>Beckman</td>
<td>101</td>
</tr>
<tr>
<td>Challon</td>
<td>125</td>
</tr>
<tr>
<td>Davis</td>
<td>105</td>
</tr>
<tr>
<td>Evans</td>
<td>225</td>
</tr>
<tr>
<td>Fagen-U</td>
<td>107</td>
</tr>
<tr>
<td>Gunter</td>
<td>422</td>
</tr>
<tr>
<td>Herman</td>
<td>233</td>
</tr>
</tbody>
</table>
General Hash Function

An $O(1)$ deterministic operation that maps all keys in a universe $U$ to a defined range of integers $[0, ..., m - 1]

- A hash:

- A compression:

Choosing a good hash function is tricky...
- Don’t create your own (yet*)
Hash Function

\[ h(k) = (k.\ firstName[0] + k.\ lastName[0]) \mod m \]

\[ h(k) = (\text{rand}() \times k.\ numPages) \mod m \]

\[ h(k) = (k.\ order\_{1st\ read\ by\ me}) \mod m \]
Hash Function

Aardvarks

Anonymous

By Jim Realman
Hash Function

Author Name Hash Function

‘J’ + ‘T’ = 28

<table>
<thead>
<tr>
<th>27</th>
<th>∅</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>∅</td>
</tr>
<tr>
<td>29</td>
<td>∅</td>
</tr>
<tr>
<td>30</td>
<td>Harry Potter</td>
</tr>
<tr>
<td>31</td>
<td>∅</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Hash Function

Author Name Hash Function

‘R’ + ‘L’ = 25

<table>
<thead>
<tr>
<th>Goosebumps</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>∅</td>
</tr>
<tr>
<td>27</td>
<td>∅</td>
</tr>
<tr>
<td>28</td>
<td>∅</td>
</tr>
<tr>
<td>29</td>
<td>∅</td>
</tr>
<tr>
<td>The Hobbit</td>
<td>30</td>
</tr>
</tbody>
</table>
Hash Function

Aardvarks
Anonymous
By Jim Truth

Author Name Hash Function

‘J’ + ‘T’ = 30

Goosebumps
∅
The Hobbit
∅
A hash collision occurs when multiple unique keys hash to the same value.
Perfect Hashing

If \( m \geq S \), we can write a *perfect* hash with no collisions.
General Purpose Hashing

In CS 225, we want our hash functions to work in general.
If $m < U$, there must be at least one hash collision.
General Purpose Hashing

By fixing $h$, we open ourselves up to adversarial attacks.
A Hash Table based Dictionary

**Client Code:**

```
1 | Dictionary<KeyType, ValueType> d;
2 | d[k] = v;
```

A **Hash Table** consists of three things:

1. A hash function
2. A data storage structure
3. A method of addressing *hash collisions*