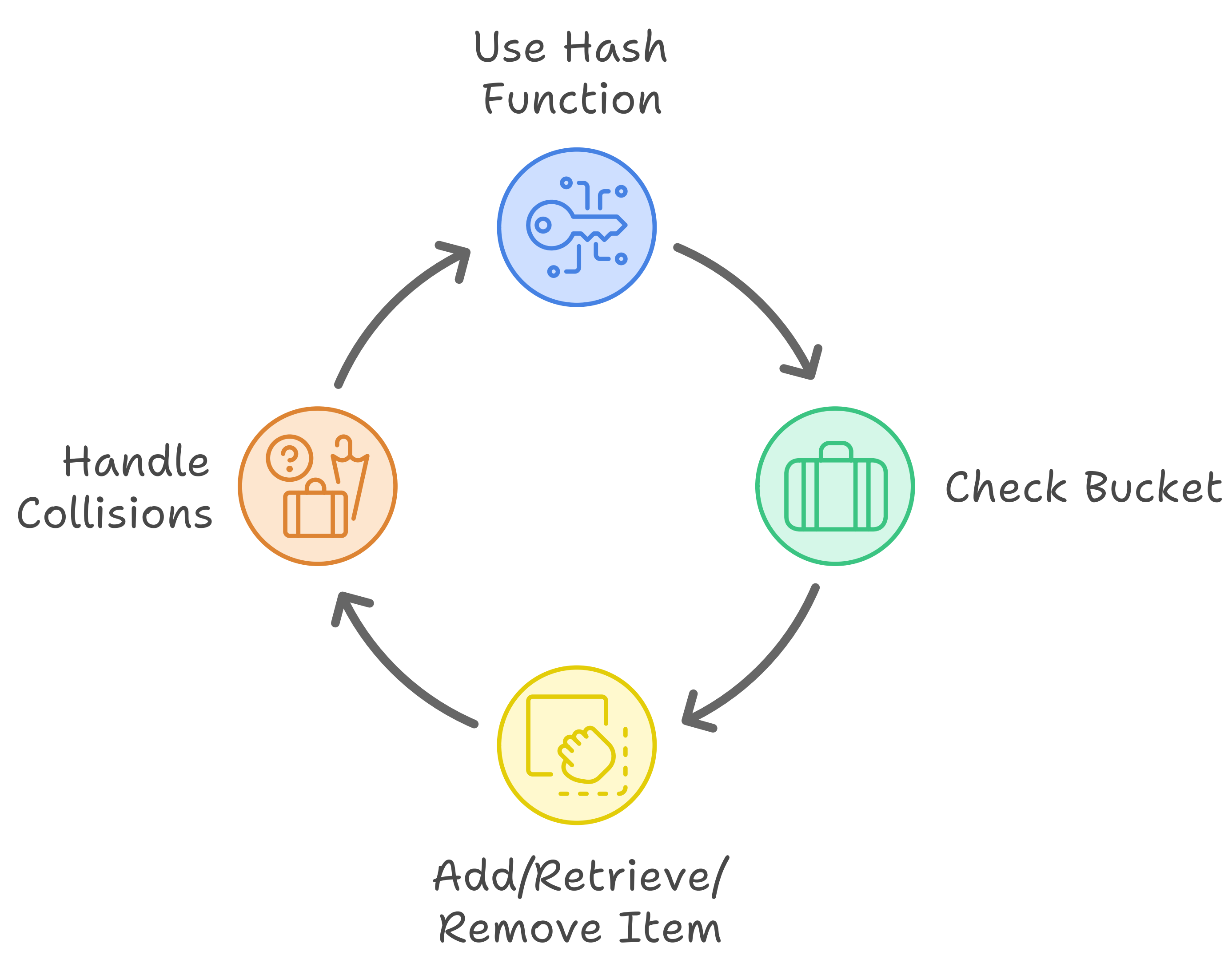
What is a Hash Map?

Imagine you have a box with many compartments (like a filing cabinet) where you can store items. Each item has a label (the key) and some information (the value). A hash map is like that box, allowing you to quickly find your items based on their labels.

Hash Map Operations Cycle



How Does It Work?

1. **Buckets**:
   - The hash map uses a list of compartments called **buckets**. Each bucket can hold one item or be empty.
   - When you want to store an item, the hash map needs to decide which bucket to use.
2. **Hashing**:
   - To find out which bucket to use for an item, we use something called a **hash function**. Think of this as a magic formula that takes the label (key) and gives us a number (index) that tells us which compartment to check.
   - For example, if your label is "apple", the hash function might say, "Put it in compartment 3."
3. **Adding Items**:
   - When you add an item, the hash map checks if the calculated bucket is empty.
   - If it's empty, it puts the item there. If it's already occupied (someone else's item is there), it uses **linear probing**—this means it checks the next compartment until it finds an empty one or finds the same key.
4. **Retrieving Items**:
   - When you want to get an item, you again use the hash function to find out which bucket to check.
   - If the item is there, great! If not, it checks the next compartment until it either finds it or hits an empty one.
5. **Removing Items**:
   - Removing works similarly: calculate the bucket index and check until you find the item or hit an empty spot

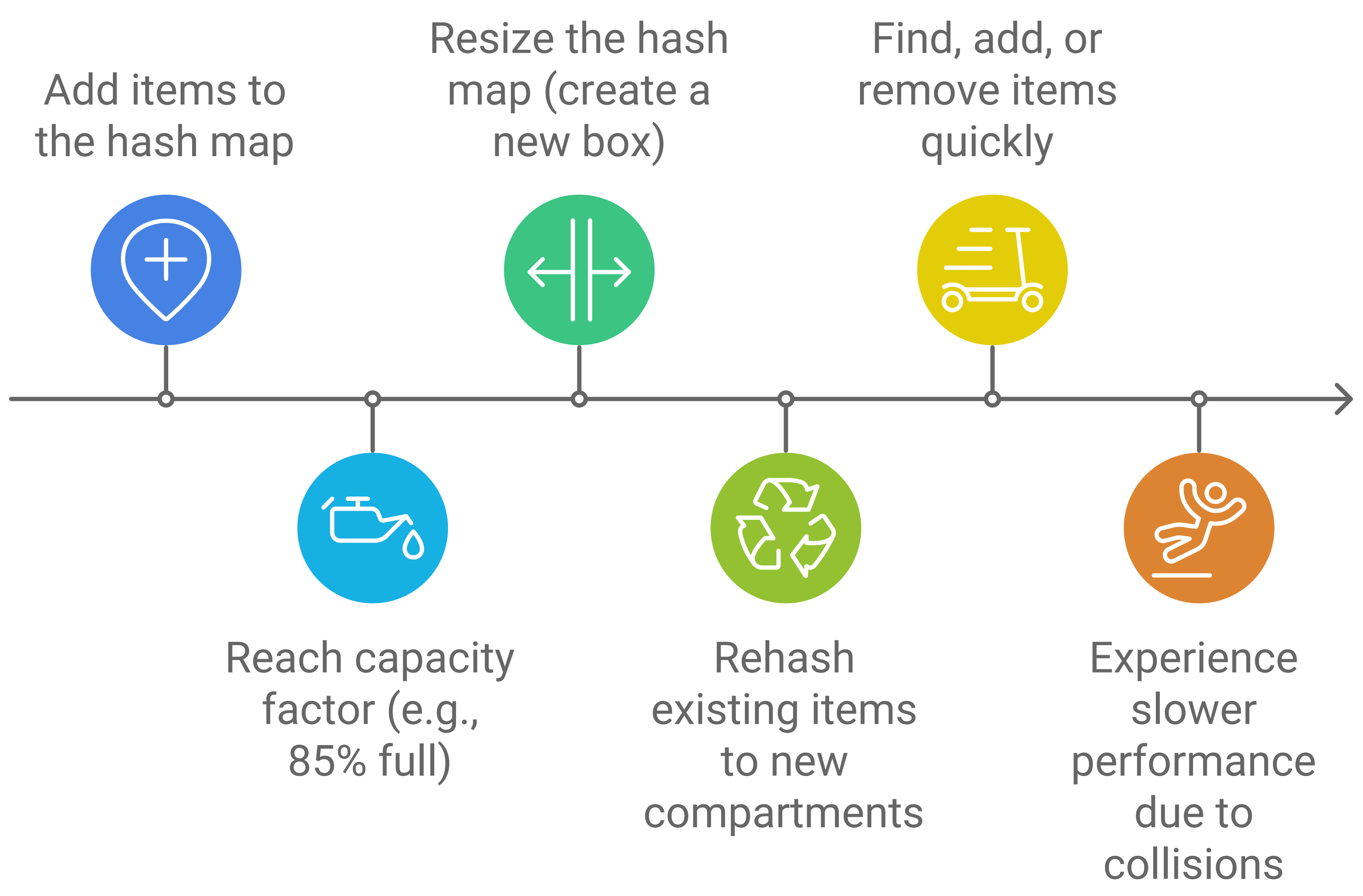
Handling Overcrowding

- As you add more items, eventually, the compartments will fill up. This is where the **capacity factor** comes in. It's like saying, "I'll start looking for more space when my box is 85% full."
- When that happens, the hash map creates a new box (resizes) with more compartments and moves all existing items into this new box using their labels again to find their new spots.

Performance

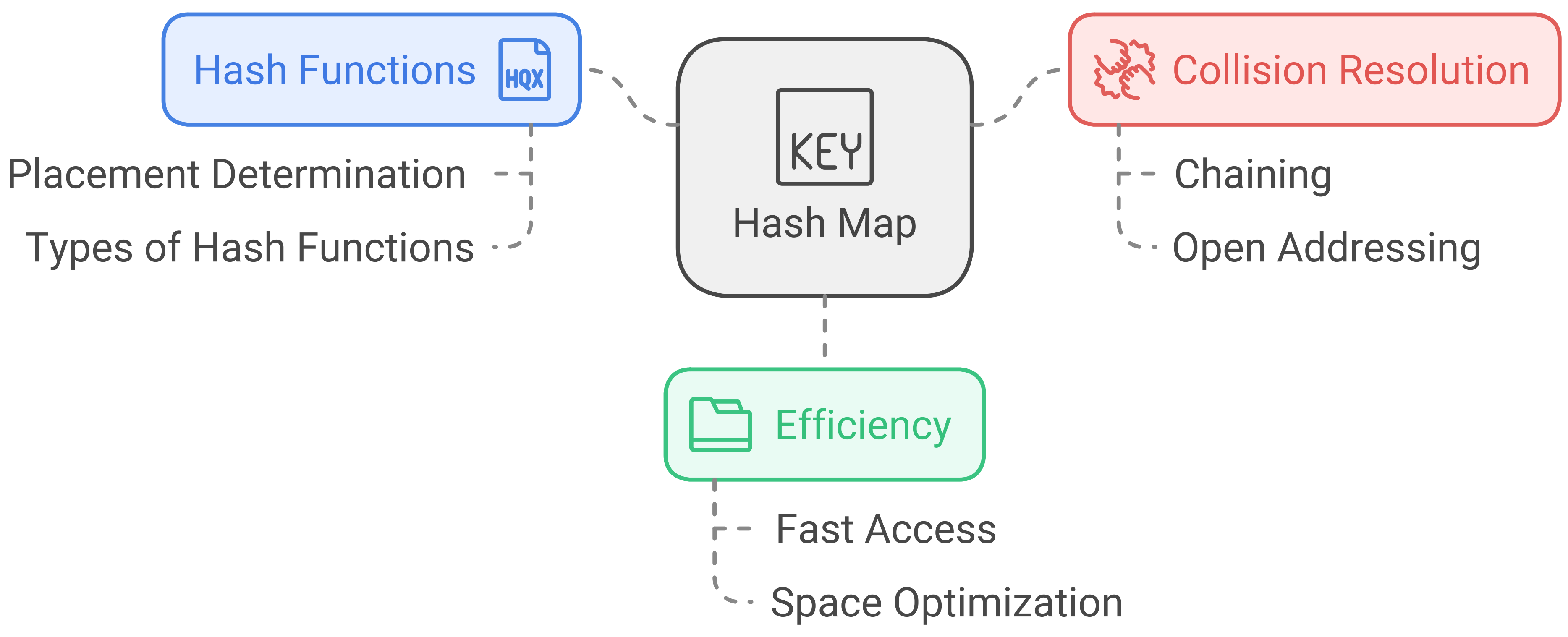
- In most cases, finding, adding, or removing items takes very little time—like reaching into your box and grabbing what you need.
- However, if too many items end up in one compartment (due to collisions), it might take longer because you have to check more compartments.

Managing Overcrowding in a Hash Map


Add items to the hash map
Reach capacity factor (e.g., 85% full)
Resize the hash map (create a new box)
Rehash existing items to new compartments
Find, add, or remove items quickly
Experience slower performance due to collisions

Summary

In summary, think of a hash map as a smart filing cabinet that uses labels to quickly find and organize items. It uses magic formulas (hash functions) to determine where each item goes and has strategies for when things get crowded or when two items want to share the same space.


Hash Functions
Placement Determination
Types of Hash Functions
Hash Map
Collision Resolution
Chaining
Open Addressing
Efficiency
Fast Access
Space Optimization