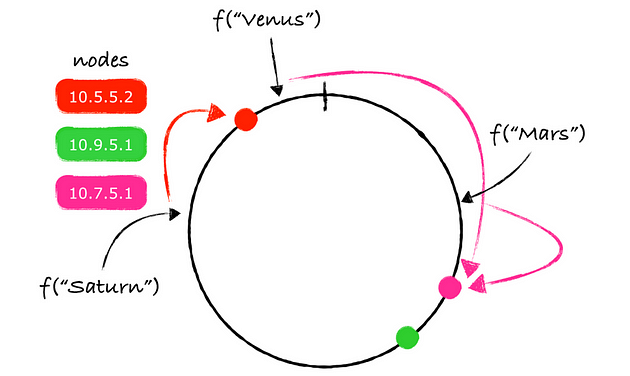
**What is Consistent Hashing? What Problem does it Solve?**

How Consistent Hashing work and how it is used in Content Delivery Networks (CDNs) and distributed caches.



Hello folks, if you are preparing for [System design interviews](https://medium.com/javarevisited/7-system-design-problems-to-crack-software-engineering-interviews-in-2023-13a518467c3e) then knowing popular System Design algorithms which is used to solve distributed system problems is mandatory. Consistent hashing is one such algorithms.

In the past, I have shared [**10 System design concepts for developers**](https://medium.com/javarevisited/top-10-system-design-concepts-every-programmer-should-learn-54375d8557a6) and in this article, I am going to share what is Consistent Hashing, how it works, where it is used and pros and cons of this popular algorithms.

But first let’s understand why do we need consistent hashing? In modern distributed systems, data needs to be distributed among multiple nodes to ensure scalability, load balancing, and fault tolerance.

However,**distributing data among nodes can be a challenging task, especially when nodes are added or removed from the system**. One solution to this problem is Consistent Hashing, a popular technique used in many distributed systems.

*Consistent Hashing* provides a scalable and flexible solution for distributing data among nodes while ensuring uniform data distribution, fault tolerance, and load balancing.

As I said, in this article, we will explore what Consistent Hashing is, how it works, and what problems it solves in distributed systems. We will also discuss some real-world examples of Consistent Hashing and its benefits and drawbacks.

By the way, if you are not a Medium member then I highly recommend you to join Medium and read great stories from great authors from real field. You can **join Medium**[**here**](https://medium.com/@somasharma_81597/membership)

**[Join Medium with my referral link - Soma](https://medium.com/@somasharma_81597/membership" \t "_blank)**

[Read this and every story from thousands of great writers on Medium). Read more from great writers on Medium like…](https://medium.com/@somasharma_81597/membership" \t "_blank)

[medium.com](https://medium.com/@somasharma_81597/membership" \t "_blank)

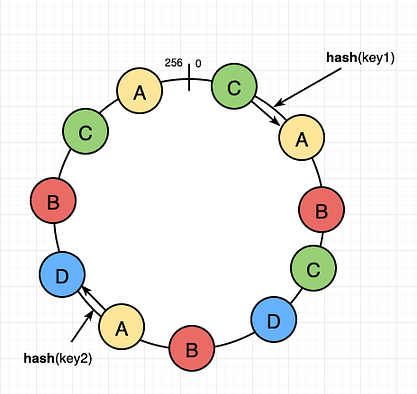
**What is Consistent Hashing? How does it work?**

As I said, Consistent hashing is a technique used in distributed systems to efficiently distribute data among multiple nodes. It is used to minimize the amount of data that needs to be transferred between nodes when a node is added or removed from the system.

The**basic idea behind consistent hashing is to use a hash function to map each piece of data to a node in the system**. Each node is assigned a range of hash values, and any data that maps to a hash value within that range is assigned to that node.

When a node is added or removed from the system,*only the data that was assigned to that node needs to be transferred to another node*. This is achieved by using a concept called virtual nodes.

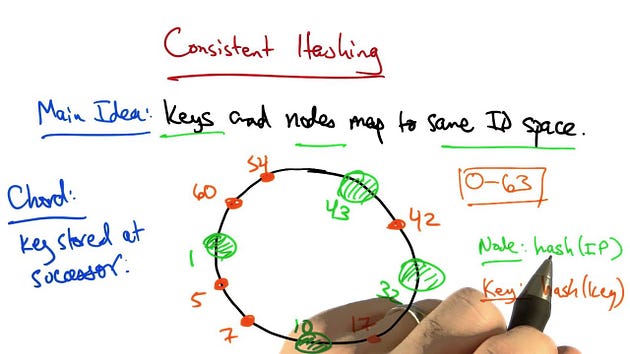
*Instead of assigning each physical node a range of hash values, multiple virtual nodes are assigned to each physical node.*



Each virtual node is assigned a unique range of hash values, and any data that maps to a hash value within that range is assigned to the corresponding physical node.

When a node is added or removed from the system, only the virtual nodes that are affected need to be reassigned, and any data that was assigned to those virtual nodes is transferred to another node.**This allows the system to scale dynamically and efficiently,** without requiring a full redistribution of data each time a node is added or removed.

In shot, consistent hashing provides a simple and efficient way to distribute data among multiple nodes in a distributed system. It is commonly used in large-scale distributed systems, such as *content delivery networks and distributed databases*, to provide high availability and scalability.

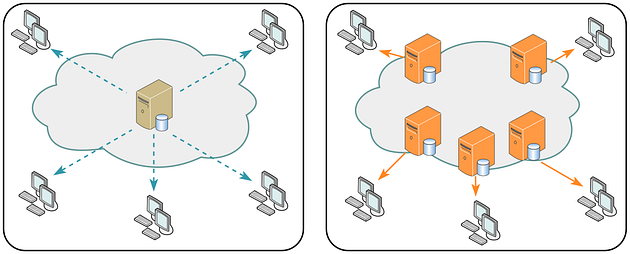


**Where is Consistent Hashing used in real world?**

Consistent Hashing is used in many distributed systems where data needs to be distributed across multiple nodes. Here are some real world examples of consistent hashing

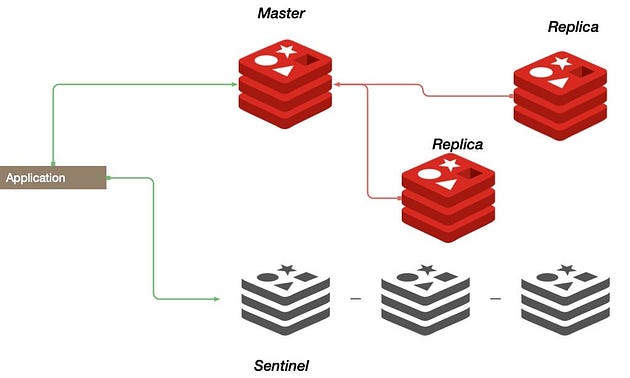
**1. Content Delivery Networks (CDNs)**

CDNs use Consistent Hashing to distribute content across multiple edge servers. Each edge server is responsible for a range of hash values, and any content that maps to a hash value within that range is served by that server.



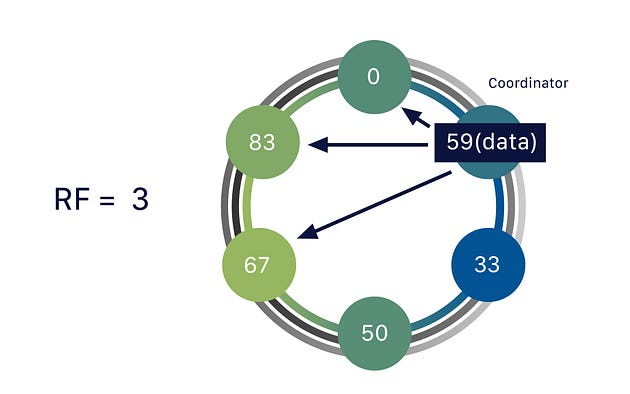
**2. Distributed Caches**

Distributed caches like Redis and Memcached use Consistent Hashing to distribute data among multiple cache nodes. Each node is responsible for a range of hash values, and any data that maps to a hash value within that range is stored in that node.



**3. Key-Value Stores**

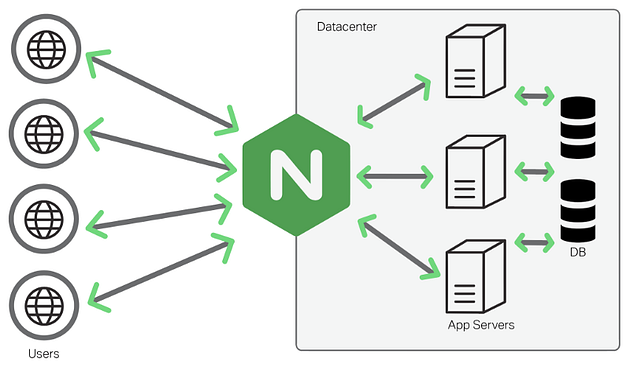
Many key-value stores like Cassandra and Riak use Consistent Hashing to distribute data among multiple nodes. Each node is responsible for a range of hash values, and any data that maps to a hash value within that range is stored in that node.



**4. Load Balancers**

Load balancers like HAProxy and Nginx use Consistent Hashing to distribute incoming requests among multiple backend servers. Each backend server is responsible for a range of hash values, and any request that maps to a hash value within that range is forwarded to that server.

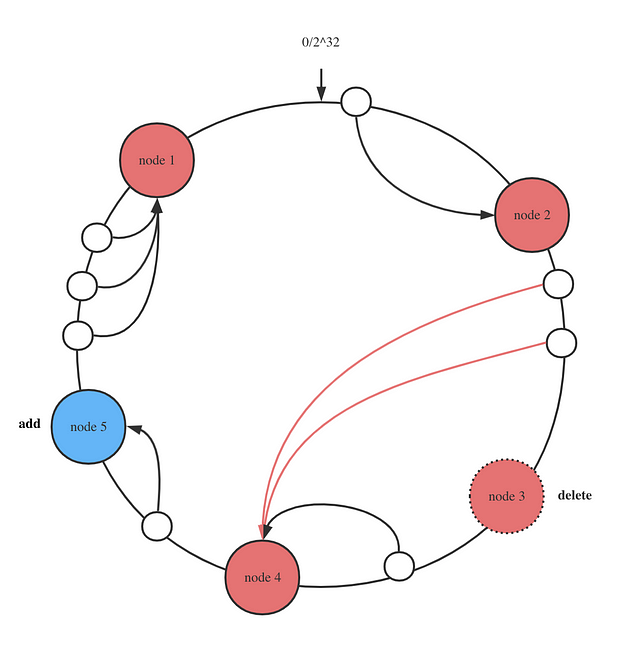
Overall, Consistent Hashing is a popular technique in many distributed systems to ensure scalability, load balancing, and fault tolerance.



**What are pros and cons of Consistent Hashing?**

Now, its time to take a look at the benefits and drawback of Consistent Hashing algorithm in distributed system. Here are the main advantages of using Consistent Hashing:

1. **Scalability:** Consistent Hashing provides a scalable solution to distribute data among multiple nodes. It allows the system to scale dynamically and efficiently, without requiring a full redistribution of data each time a node is added or removed.
2. **Load Balancing:** Consistent Hashing ensures a uniform distribution of data across nodes. Each node is responsible for a unique range of hash values, and any data that maps to a hash value within that range is assigned to that node. This ensures that the workload is distributed evenly among all the nodes, thereby avoiding hotspots.
3. **Fault Tolerance:** In case of a node failure, only the virtual nodes that are affected need to be reassigned, and any data that was assigned to those virtual nodes is transferred to another node. This allows the system to handle node failures gracefully, without losing data or affecting the performance.



Here are the main drawbacks of Consistent Hashing:

1. **Hash Function Collisions:** Consistent Hashing relies heavily on the hash function used to map data to nodes. In case of hash function collisions, the data can be distributed unevenly among nodes, leading to hotspots and affecting the performance.
2. **Overhead:**Consistent Hashing requires additional overhead to maintain the virtual nodes and the mapping of data to nodes. This can add to the complexity and cost of the system.
3. **Data Migration:**When a node is added or removed from the system, the data that was assigned to that node needs to be transferred to another node. This can be a time-consuming and resource-intensive process, especially for large datasets.

Overall, Consistent Hashing provides a simple and efficient way to distribute data among multiple nodes in a distributed system. While it has some limitations and overhead, the benefits of scalability, load balancing, and fault tolerance make it a popular choice in many large-scale distributed systems.

**Conclusion**

That’s all about **what is consistent hashing, what problem it solve, and how it works?**We have also seen real world scenarios where consistent hashing is used like load balancing and distributed caching.

In short, Consistent Hashing is a powerful technique used in many distributed systems to solve the challenge of distributing data among multiple nodes.

It provides a scalable and flexible solution for adding or removing nodes from a distributed system while ensuring uniform data distribution, fault tolerance, and load balancing. While Consistent Hashing provides many benefits, it also has some drawbacks, such as increased complexity and potential for data imbalance.

Nevertheless, Consistent Hashing remains a crucial technique in distributed systems and is essential for ensuring scalability, fault tolerance, and performance. It’s also an important algorithm to remember for system design interview, so make sure you understand and remember how consistent hashing works.