20. Merkle Trees

Let's learn about Merkle trees and their usage.

We'll cover the following

Background#

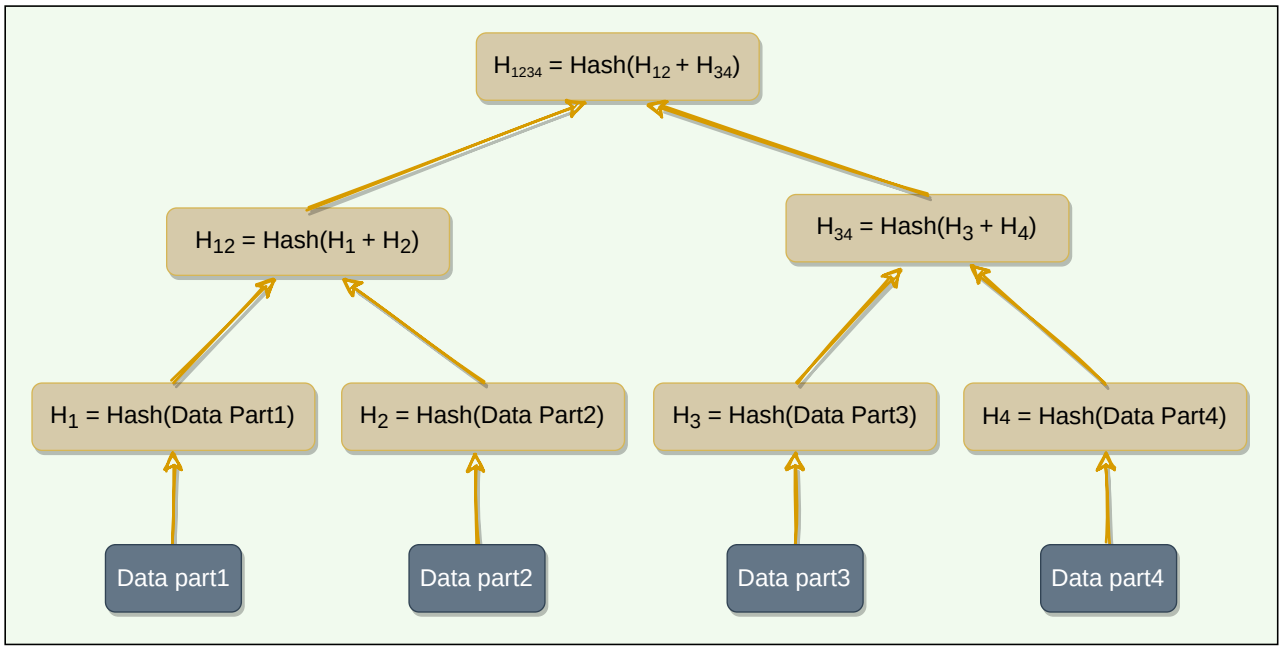
As we saw in the previous lesson (https://www.educative.io/collection/page/5668639101419520/5559029852536832/5978407785988096), Read Repair removes conflicts while serving read requests. But, if a replica falls significantly behind others, it might take a very long time to resolve conflicts. It would be nice to be able to automatically resolve some conflicts in the background. To do this, we need to quickly compare two copies of a range and figure out exactly which parts are different. In a distributed environment, how can we quickly compare two copies of a range of data residing on two different replicas and figure out exactly which parts are different?

Definition#

A replica can contain a lot of data. Naively splitting up the entire range to calculate checksums for comparison, is not very feasible; there is simply too much data to be transferred. Instead, we can use Merkle trees to compare replicas of a range.

Solution#

A Merkle tree is a binary tree of hashes, where each internal node is the hash of its two children, and each leaf node is a hash of a portion of the original data.



Merkle tree

Comparing Merkle trees is conceptually simple:

1. Compare the root hashes of both trees.
2. If they are equal, stop.
3. Recurse on the left and right children.

Ultimately, this means that replicas know exactly which parts of the range are different, but the amount of data exchanged is minimized. The principal advantage of a Merkle tree is that each branch of the tree can be

checked independently without requiring nodes to download the entire tree or the entire data set. Hence, Merkle trees minimize the amount of data that needs to be transferred for synchronization and reduce the number of disk reads.

The disadvantage of using Merkle trees is that many key ranges can change when a node joins or leaves, at which point the trees need to be recalculated.

Examples#

For anti-entropy and to resolve conflicts in the background, **Dynamo** uses Merkle trees.

← Back

19. Read Repair

Next →

System Design Basics

Mark as Completed

Report an Issue