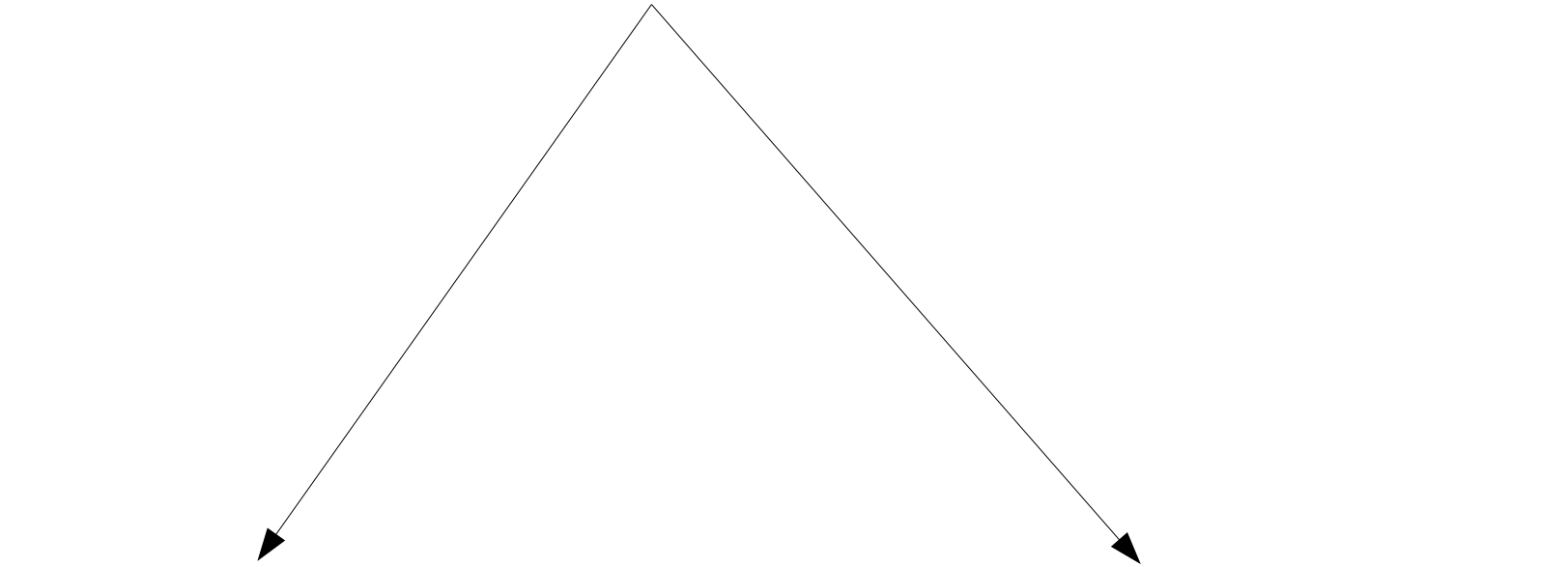
Authenticated Data Structures for Blockchain

Alexander Chepurnoy

Ergo Platform &
IOHK Research

Hash function: $\{0,1\}^* \rightarrow \{0,1\}^L$



Standard hash function

(Java's hashCode(), Linux' sum)

Cryptographic hash function

SHA-256, Keccak , Blake2b, ...

Difference?

Collisions!

A collision is H(x) = H(y), for x ≠y

- For a standard hash functions, collisions are okay (should be minimized, but aviolation is about efficiency usually)
- For a cryptographic hash function, finding collision should be impossible even for an active attacker!

Formal Definition

- A game between a challenger and an adversary
- Adversary is not limited except of a computational class of algorithms used by him (usually probabilistic polynomial-time)
- **No polynomially bound Adversary can win the game with non-negligible probability**

Formal Definition

- A game between a challenger and an adversary
- Adversary is not limited except of a computational class of algorithms used by him (usually probabilistic polynomial-time)
- **No polynomially bound Adversary can win the game with non-negligible probability**

Hash Function: Formal Definition

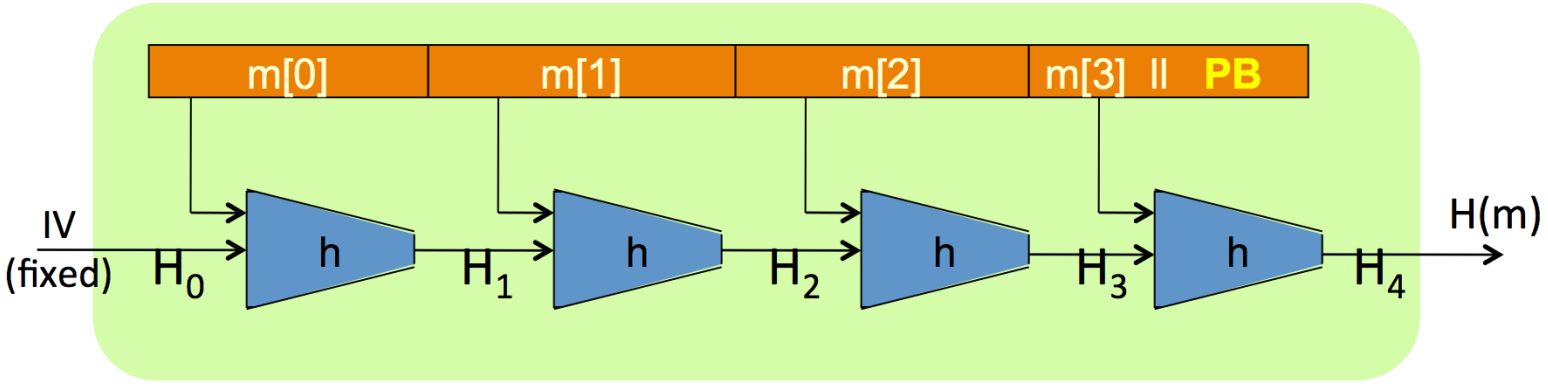
$s \leftarrow Gen(L)$
$H^s(x)$: $\{0,1\}^L$

Collision-resistance game H-Coll:
1. $s \leftarrow Gen(L)$
2. Adversary: x, x'
3. If $H^s(x) = H^s(x')$ & $x \neq x'$ return 1 else 0

For any PPT adversary, **Pr[H-Coll(L)] ≤ negl(L)**

Hash Function: Construction

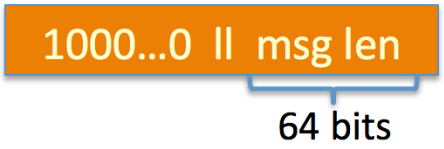
The Merkle-Damgard iterated construction



Given $h: T \times X \longrightarrow T$ (compression function)

we obtain $H: X^{\leq L} \longrightarrow T$. H_i - chaining variables

PB: padding block

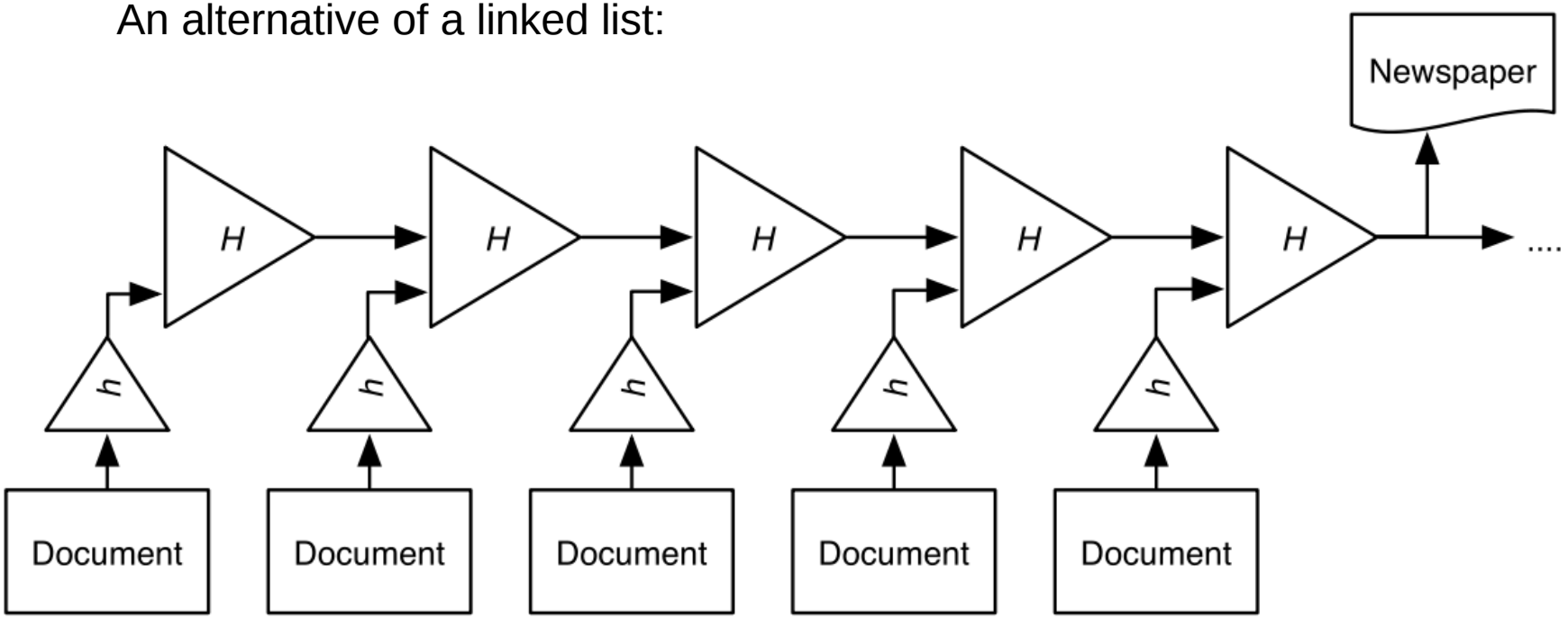


If no space for PB
add another block

Dan Boneh

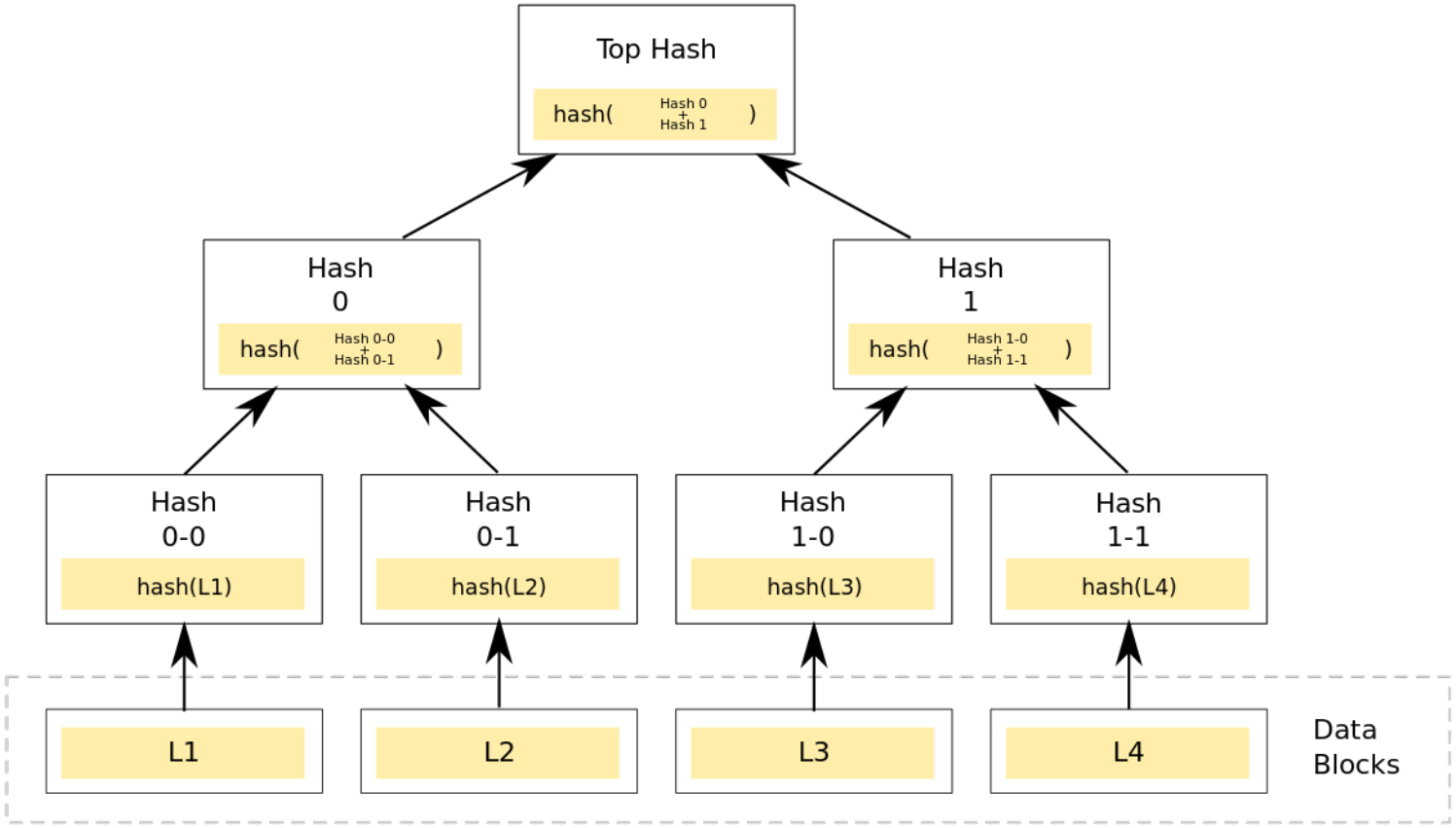
HashChain

An alternative of a linked list:

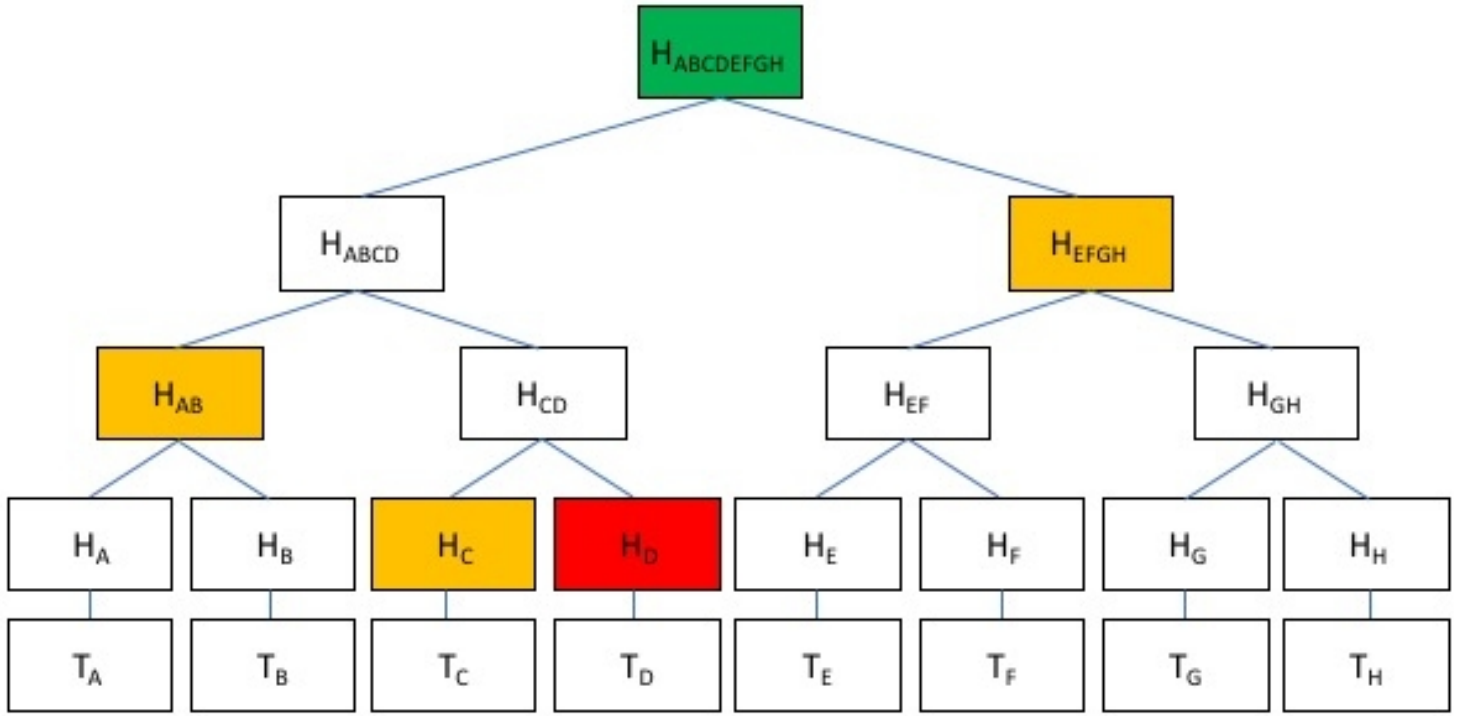


Merkle Tree

An alternative of an ideally balanced tree, log-sized proofs:

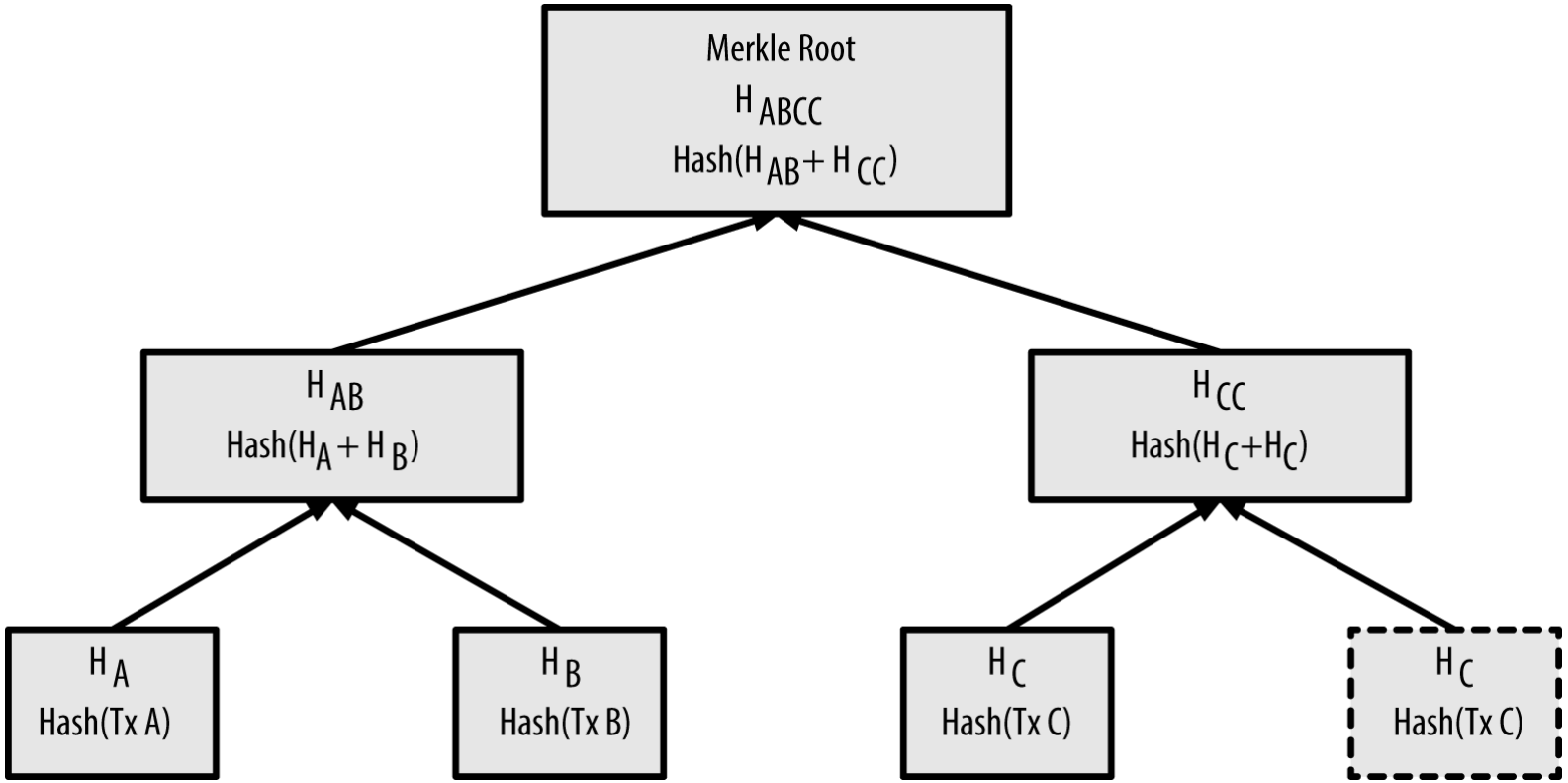


Merkle (Membership) Proof



Merkle Tree in Bitcoin

Not so ideal input:



Problems With Bitcoin’s Tree (1/2)

- Repeating leafs → repeating proofs!

  Merkle.cpp:

  “WARNING! If you're reading this because you're learning about crypto and/or designing a new system that will use merkle trees, keep in mind that the following merkle tree algorithm has a serious flaw related to duplicate txids, resulting in a vulnerability (CVE-2012-2459).”

Problems With Bitcoin’s Tree (2/2)

- No number of elements or height of a tree included.
- Merkle tree is not collision-resistant then
- Probably not a problem for Bitcoin, due to constraints on a leaf, but don’t use Bitcoin’s tree in your projects!
- If you want to avoid passing a number of elements (or a height) along with the root hash, use **domain separation**.

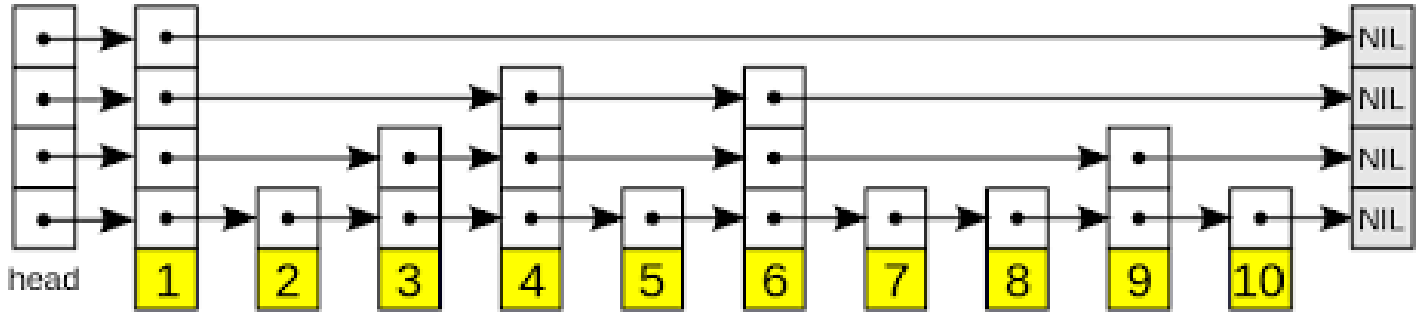
Merkle Tree

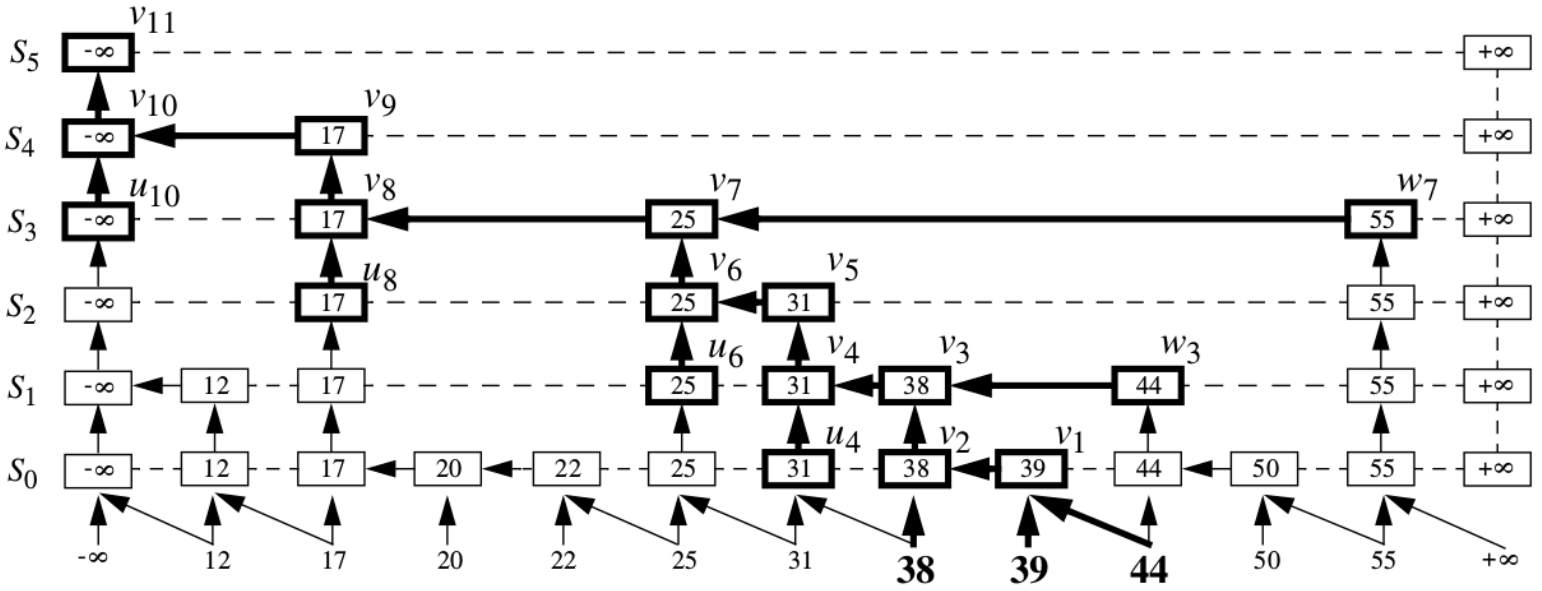
- For a static unordered set
- Simple, but easy to get wrong

Going further

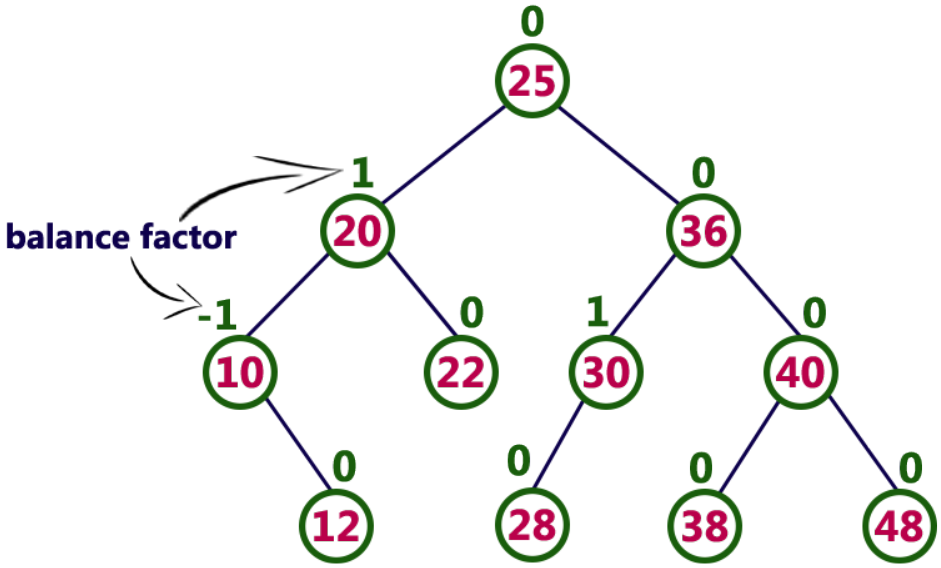
- Dynamic structures
- Ordered key → value structures
- Non-membership proofs
- Efficient

Authenticated Skiplist





AVL Tree



Questions?