Merkle trees

How Bitcoin works?

Contect

A class in data structures

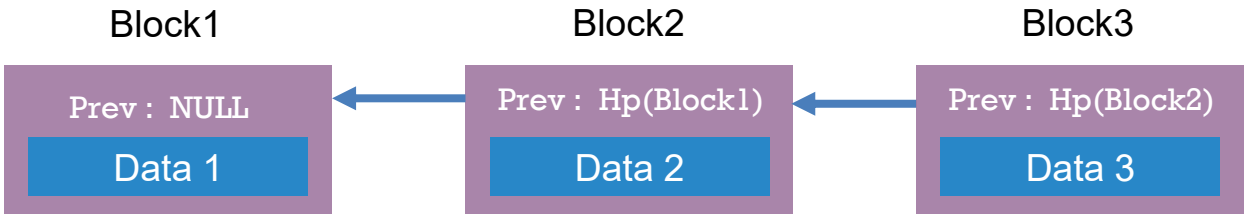
- Hash pointers
- Blockchain

Content

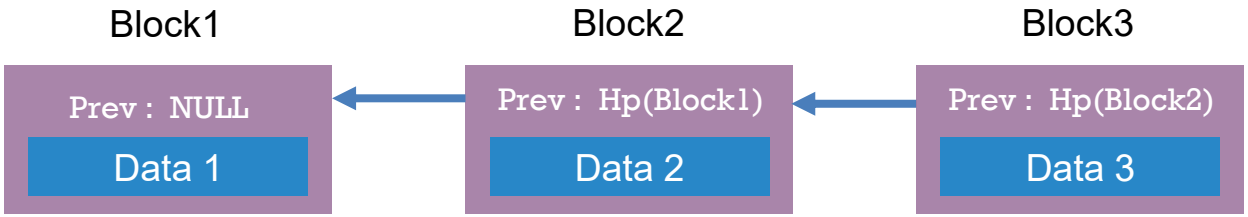
A class in data structures

- Hash pointers
- Blockchain
- Merkle trees

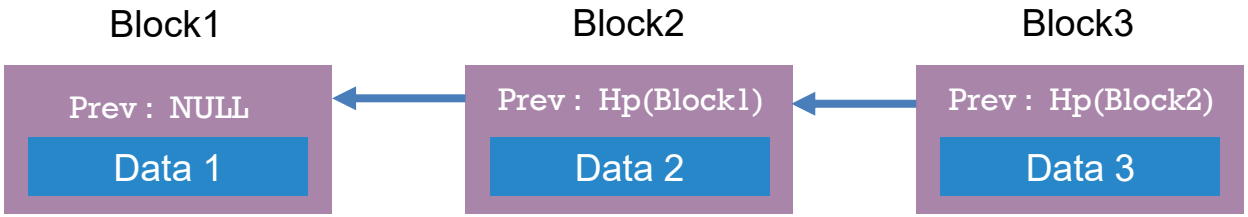
One weakness



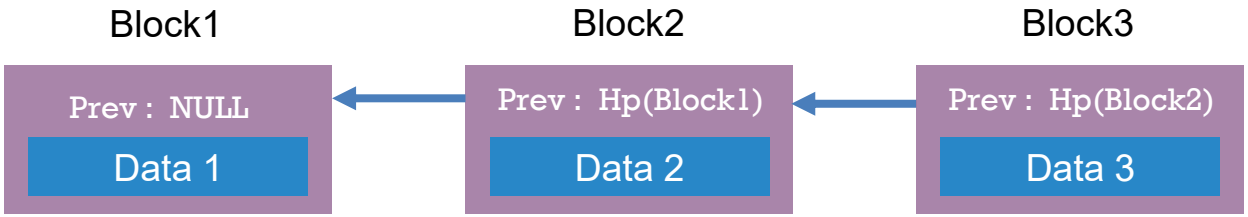
One weakness

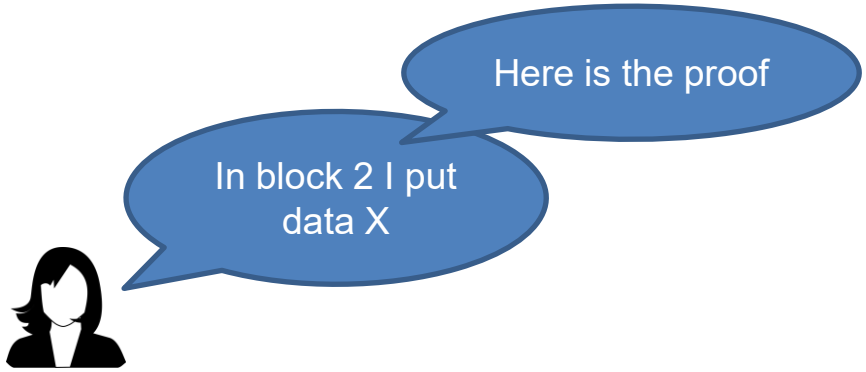


One weakness

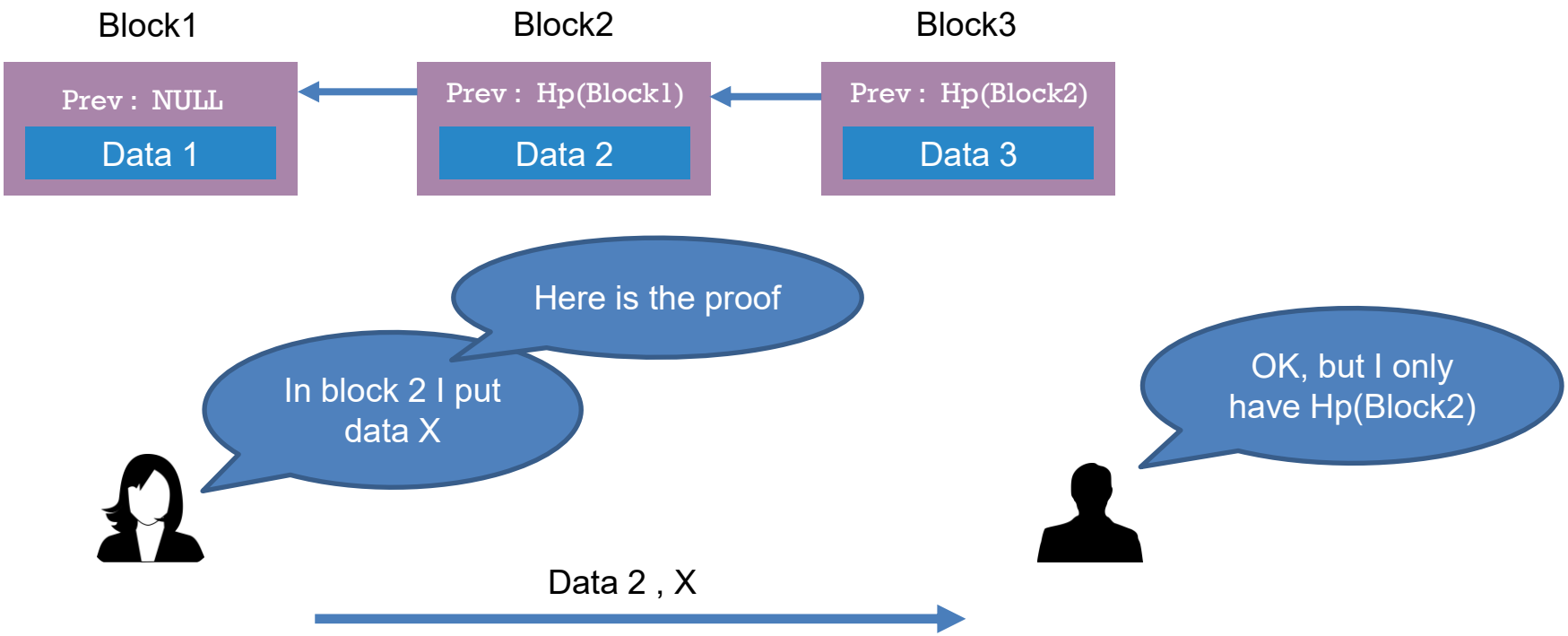


One weakness

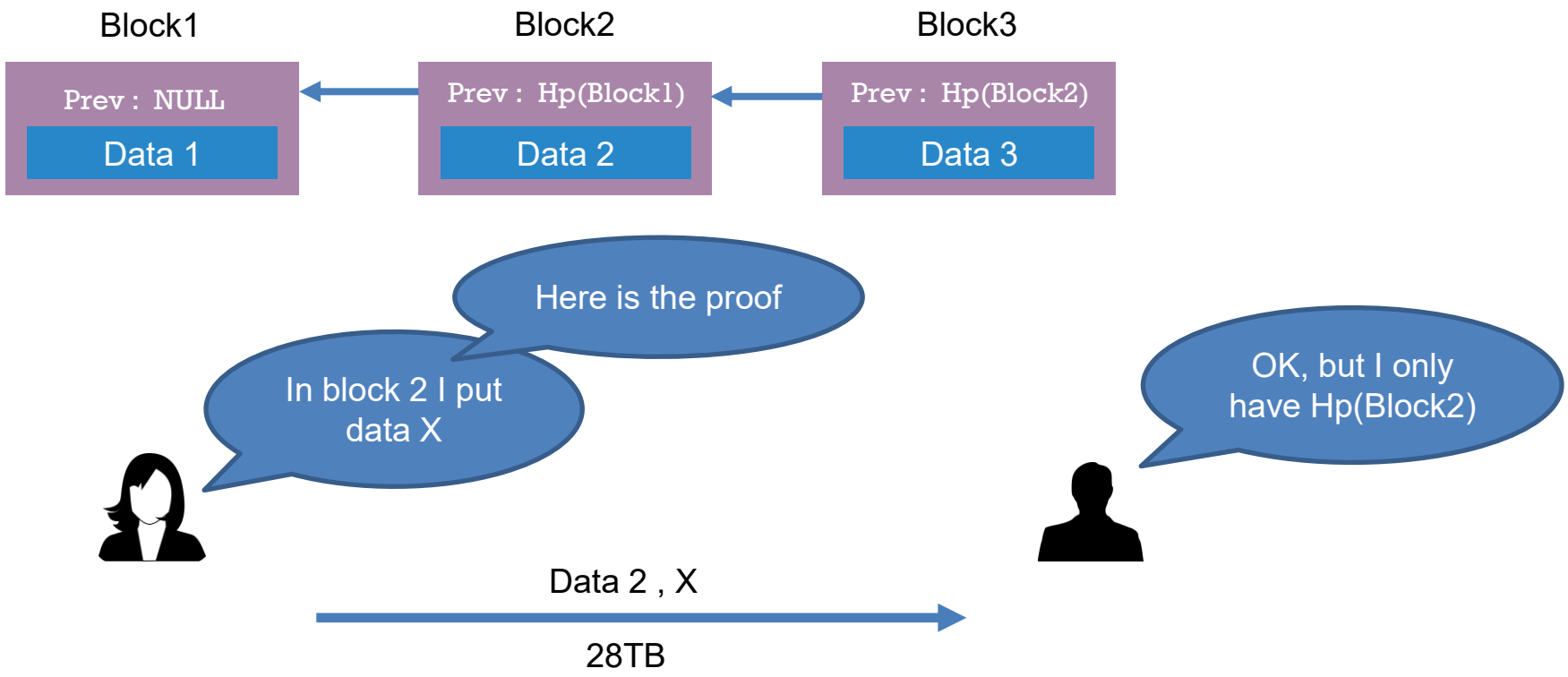




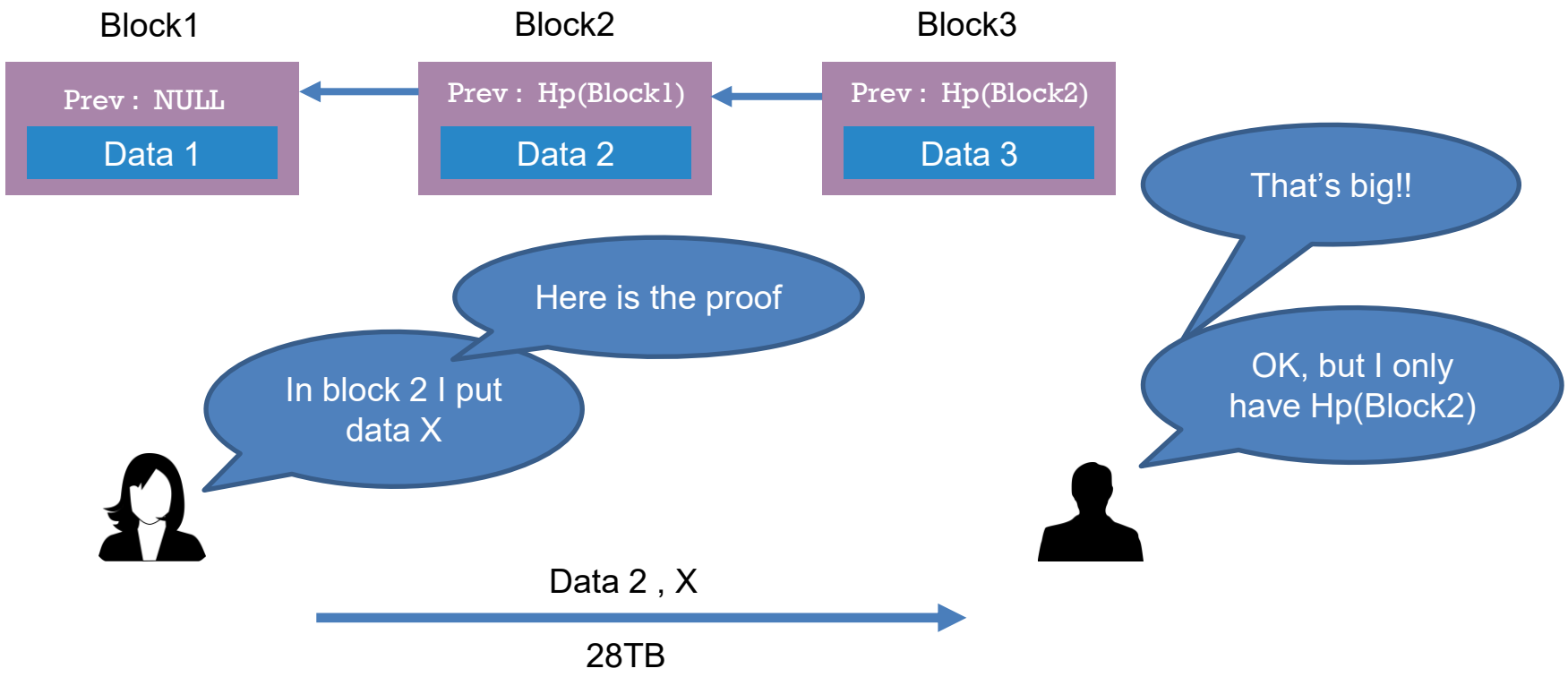
One weakness



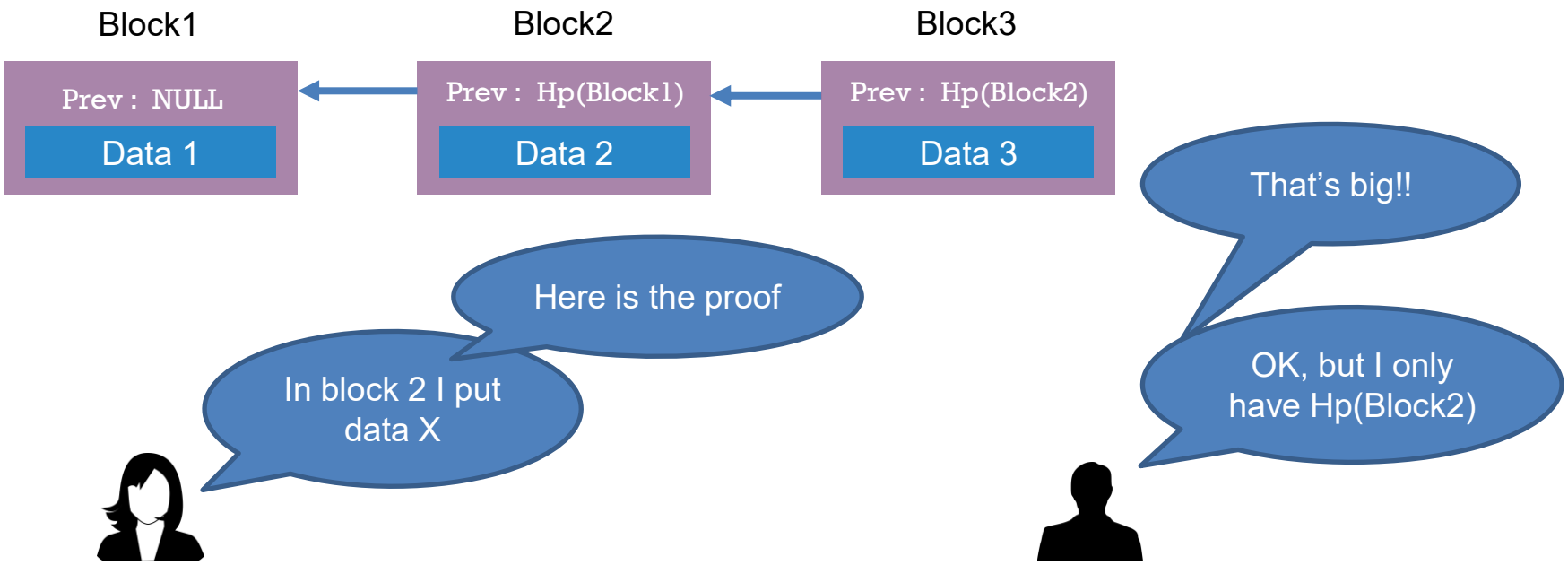
One weakness



One weakness



One weakness

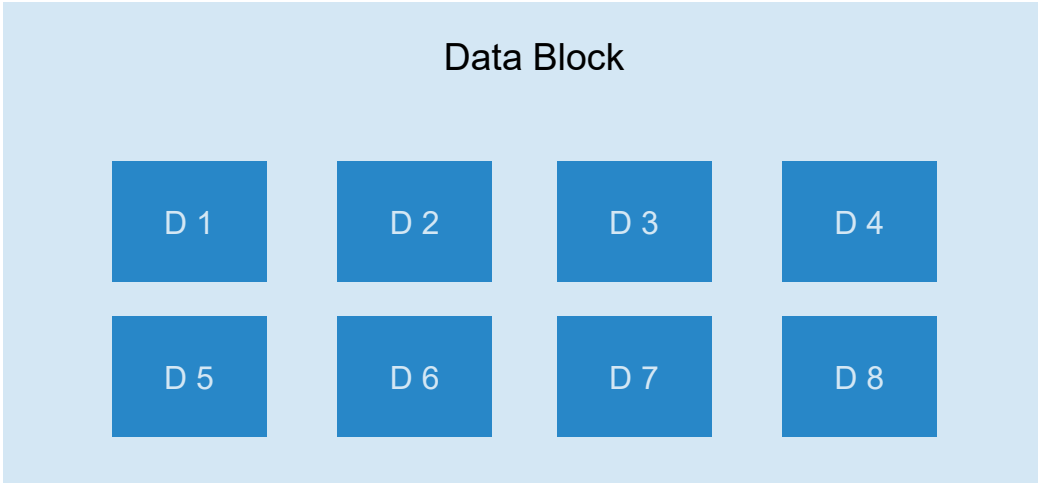


How to make the proof more efficient???

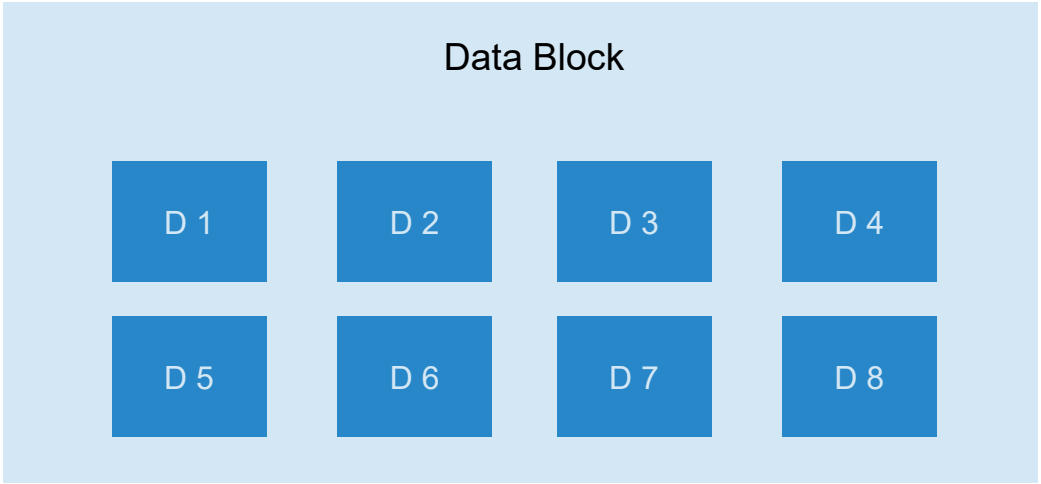
Merkle trees

What are we trying to do?

What are we trying to do?

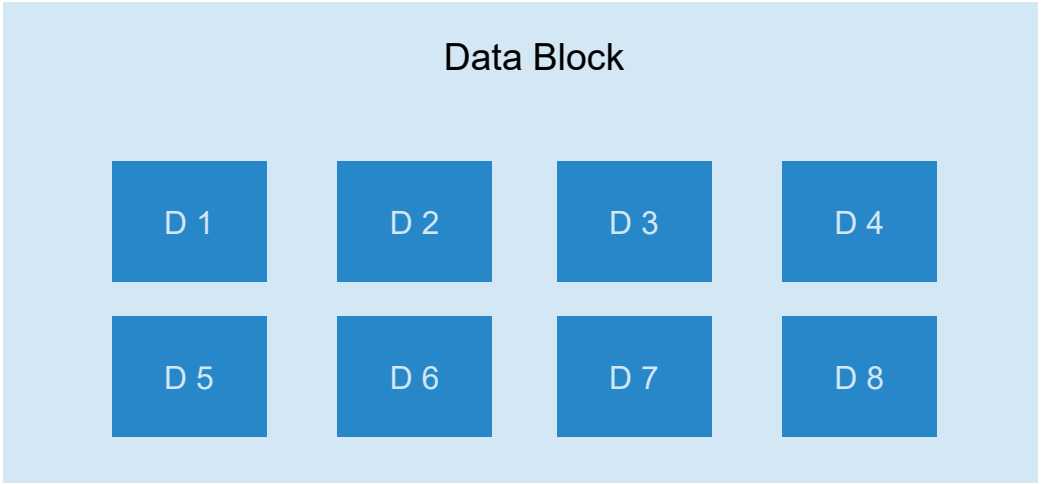


What are we trying to do?



H(Data Block) + Di allows veryfing that Di belongs to Data Block

What are we trying to do?



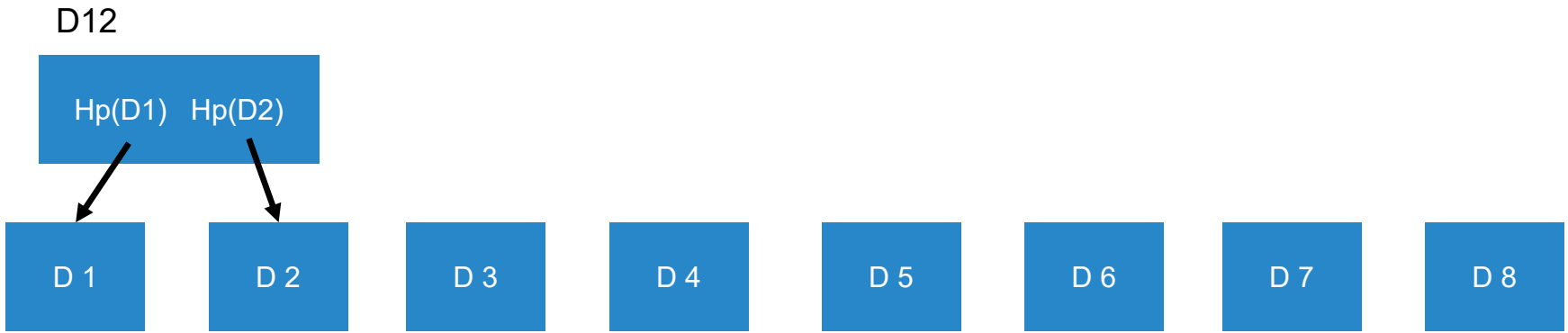
H(Data Block) + Di allows veryfing that Di belongs to Data Block

Can we acheive this?

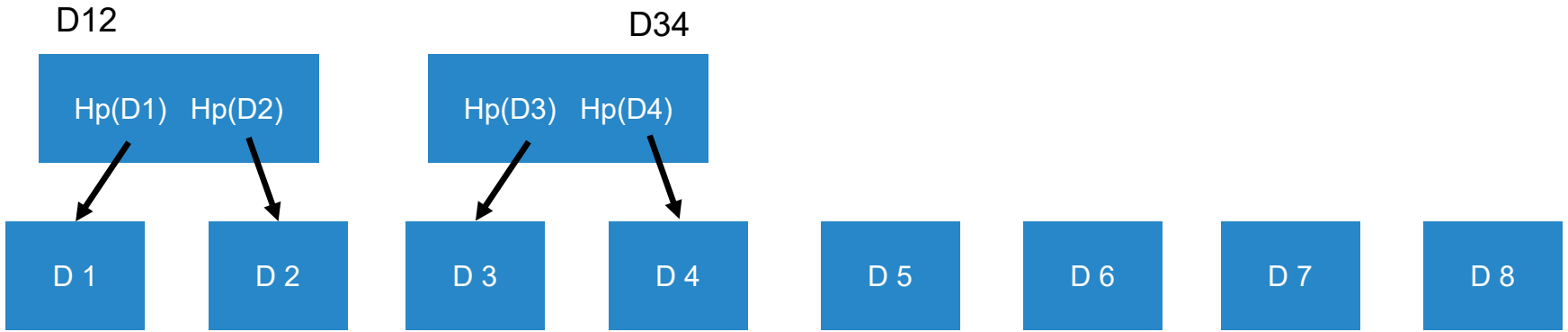
Merkle tree

Merkle tree

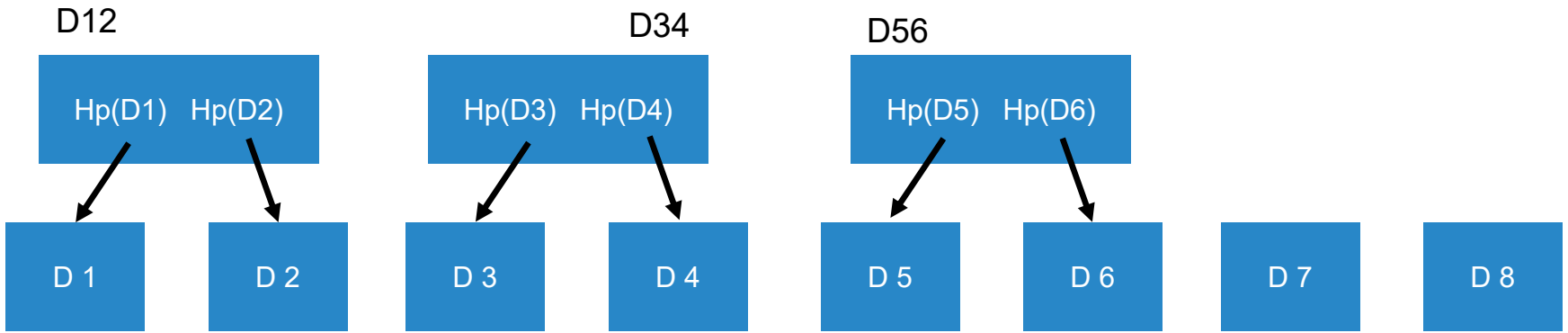
Merkle tree



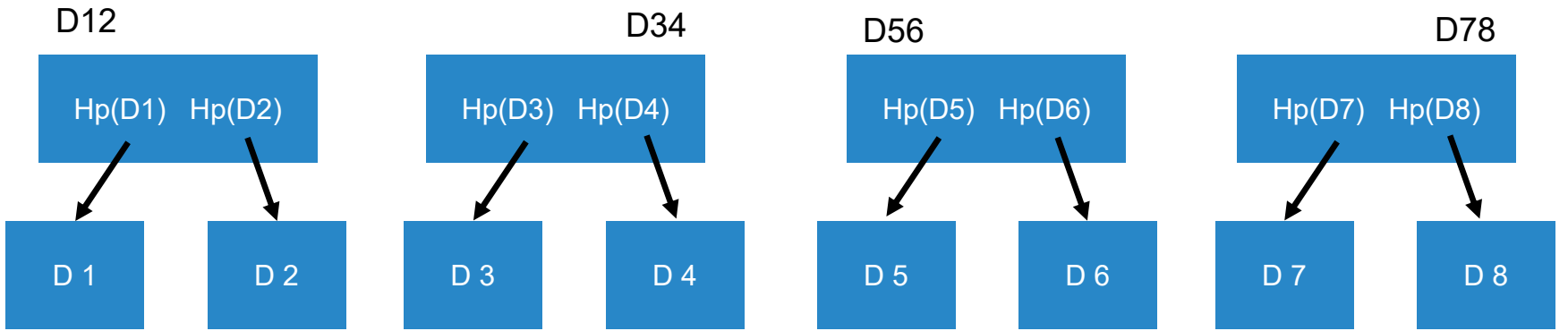
Merkle tree



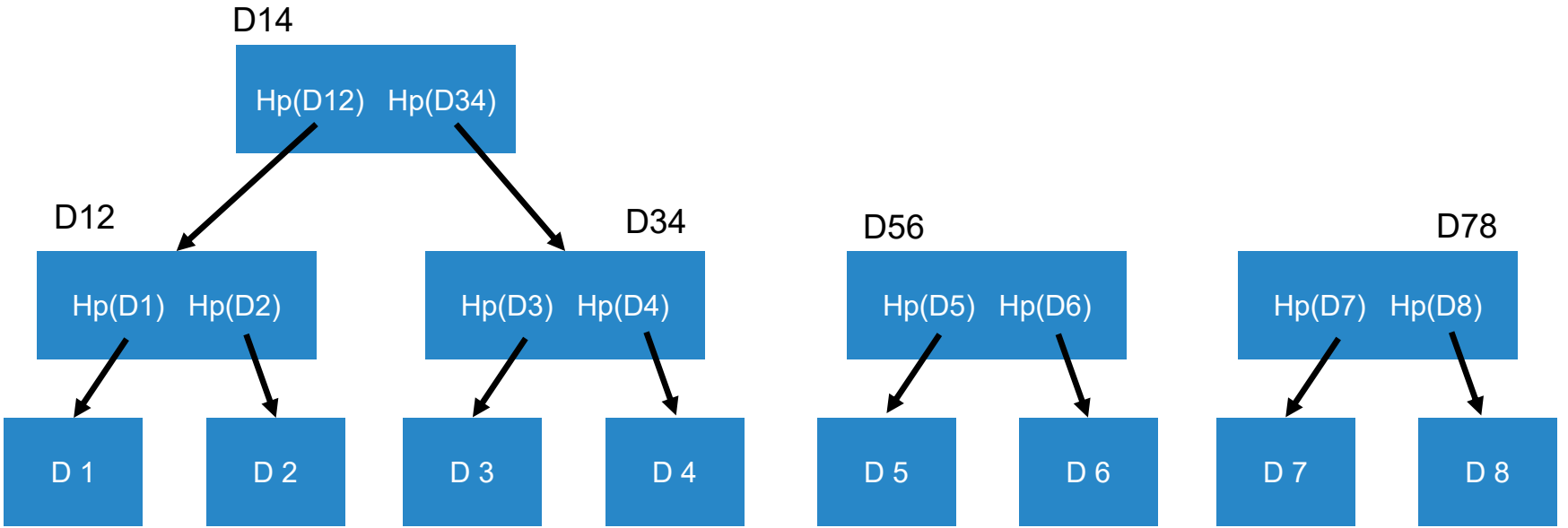
Merkle tree



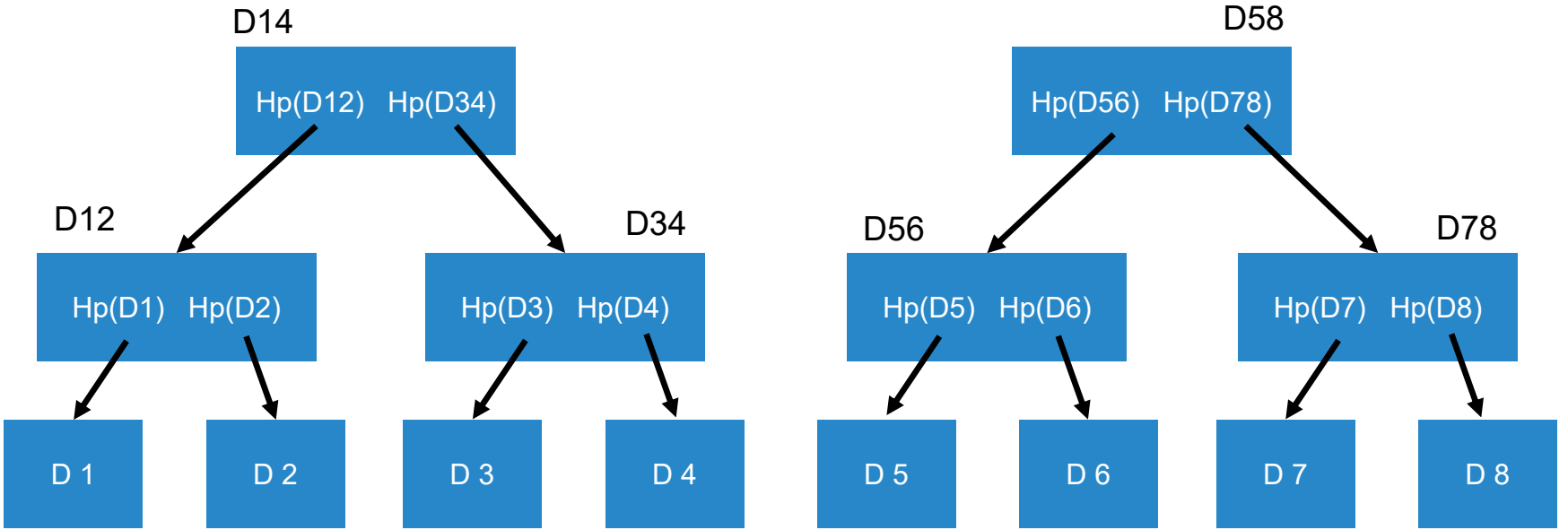
Merkle tree


D12
Hp(D1) Hp(D2)
D 1
D 2
D34
Hp(D3) Hp(D4)
D 3
D 4
D56
Hp(D5) Hp(D6)
D 5
D 6
D78
Hp(D7) Hp(D8)
D 7
D 8

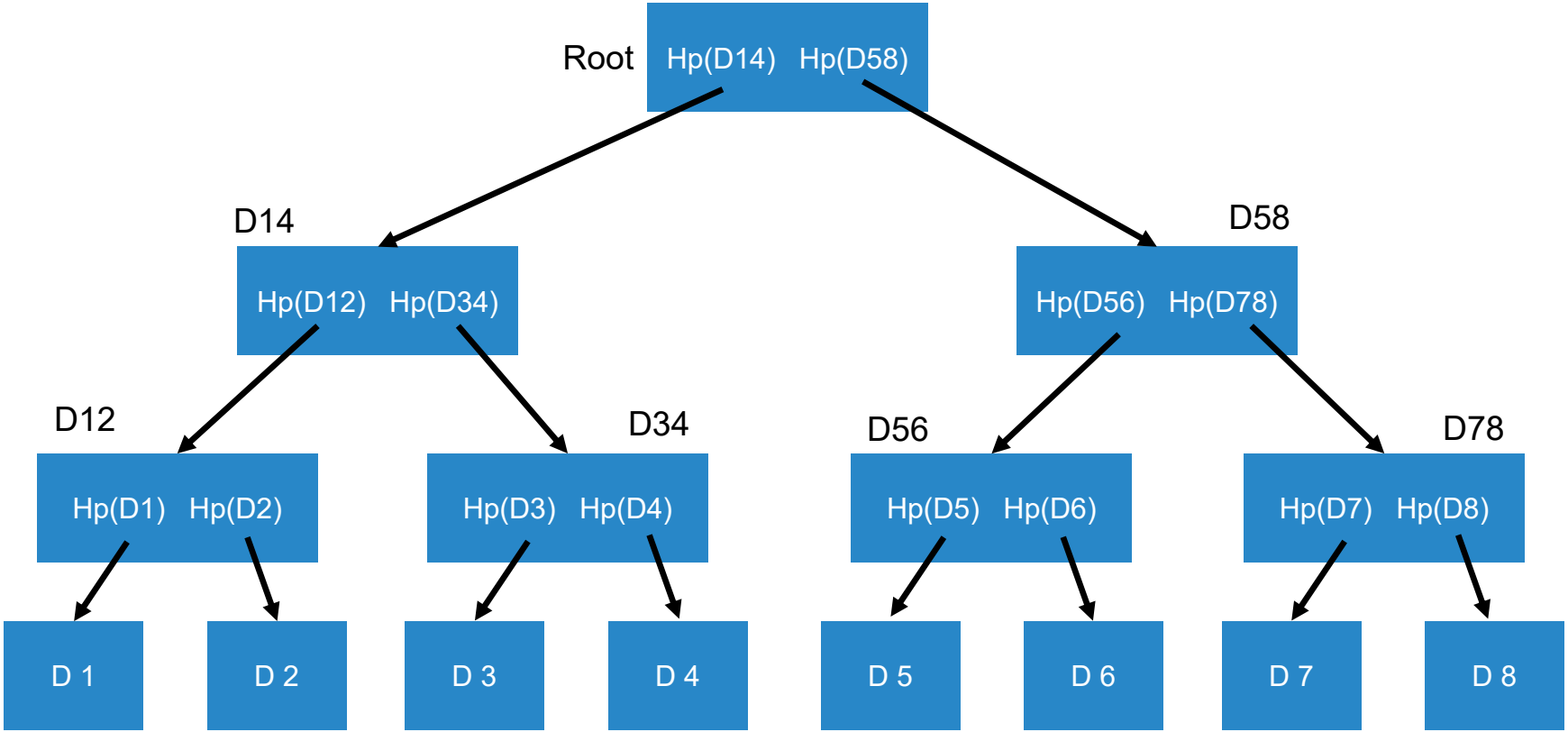
Merkle tree


D14
Hp(D12) Hp(D34)
D12
Hp(D1) Hp(D2)
D34
Hp(D3) Hp(D4)
D56
Hp(D5) Hp(D6)
D78
Hp(D7) Hp(D8)
D 1
D 2
D 3
D 4
D 5
D 6
D 7
D 8

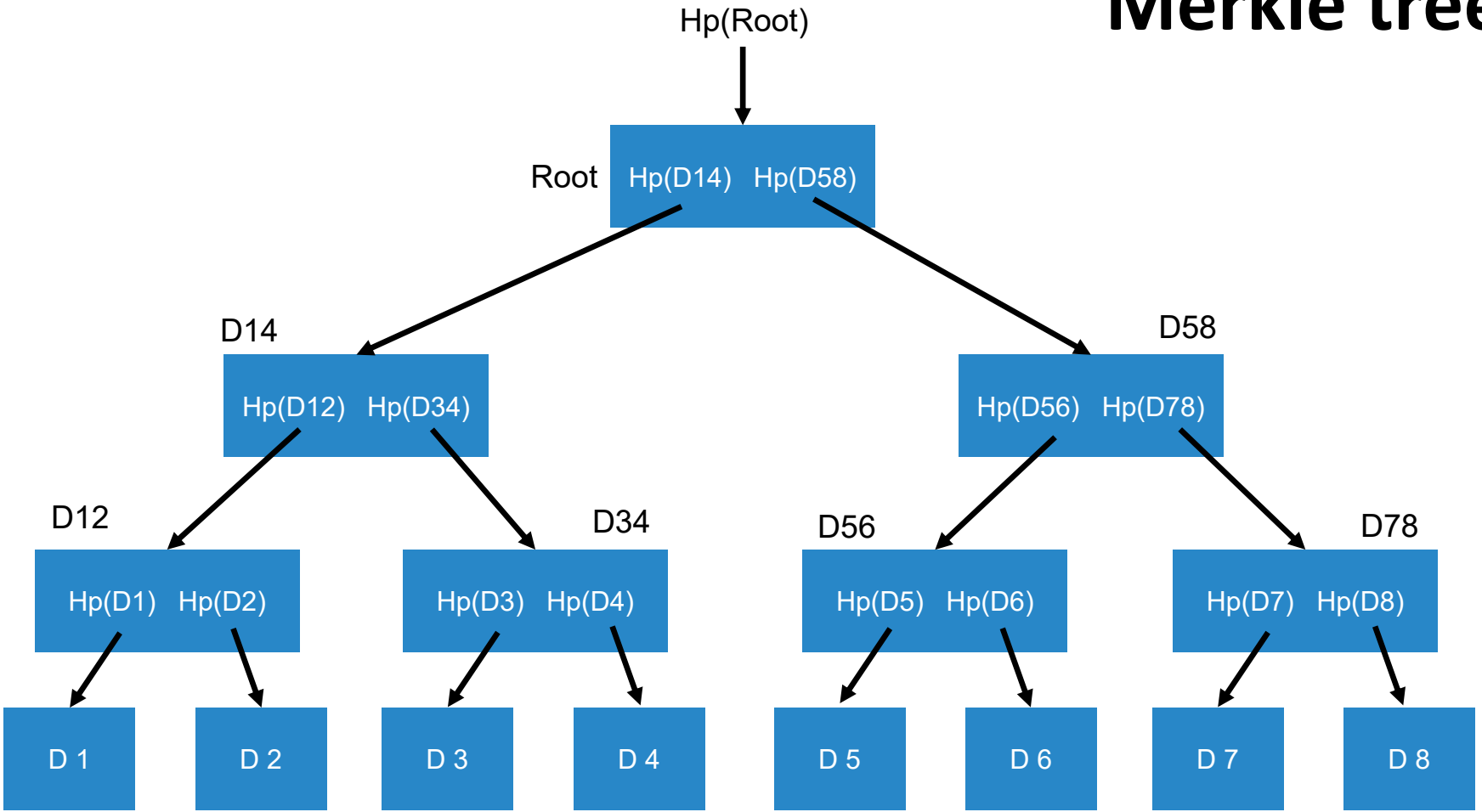
Merkle tree


D14
Hp(D12) Hp(D34)
D12
Hp(D1) Hp(D2)
D34
Hp(D3) Hp(D4)
D 1
D 2
D 3
D 4
D58
Hp(D56) Hp(D78)
D56
Hp(D5) Hp(D6)
D78
Hp(D7) Hp(D8)
D 5
D 6
D 7
D 8

Merkle tree


Root
Hp(D14) Hp(D58)
D14
Hp(D12) Hp(D34)
D58
Hp(D56) Hp(D78)
D12
Hp(D1) Hp(D2)
D34
Hp(D3) Hp(D4)
D56
Hp(D5) Hp(D6)
D78
Hp(D7) Hp(D8)
D 1
D 2
D 3
D 4
D 5
D 6
D 7
D 8

Merkle tree

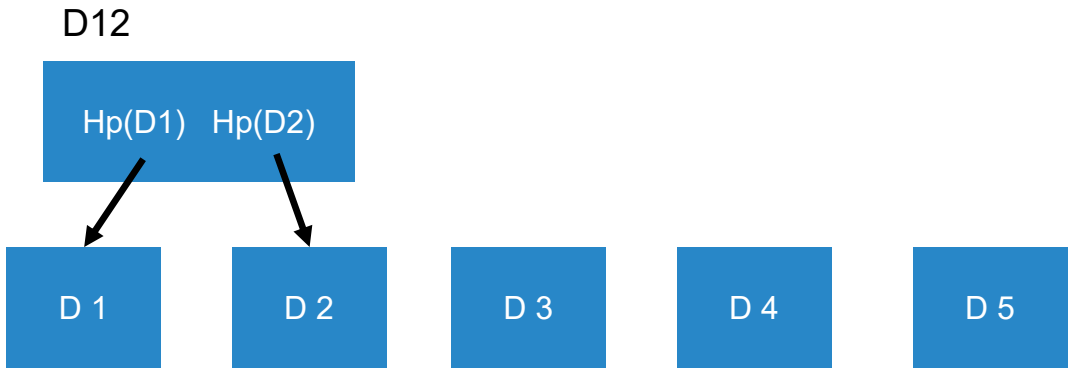

Hp(Root)
Root
Hp(D14) Hp(D58)
D14
Hp(D12) Hp(D34)
D58
Hp(D56) Hp(D78)
D12
Hp(D1) Hp(D2)
D34
Hp(D3) Hp(D4)
D56
Hp(D5) Hp(D6)
D78
Hp(D7) Hp(D8)
D 1
D 2
D 3
D 4
D 5
D 6
D 7
D 8

Merkle tree

If I don't have 2^n data?

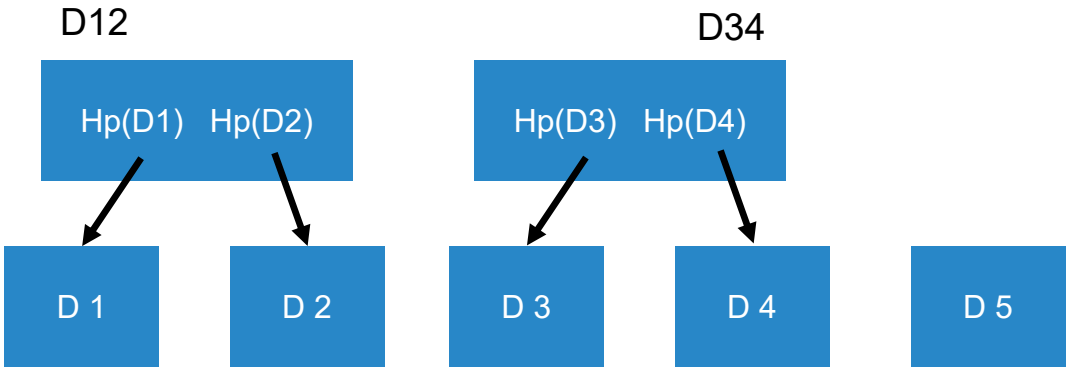
Merkle tree

If I don't have 2^n data?



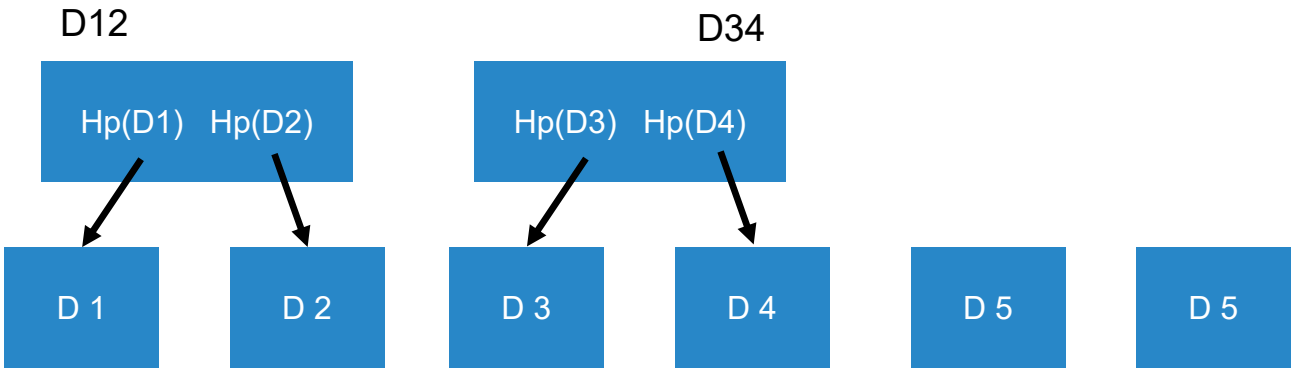
Merkle tree

If I don't have 2^n data?



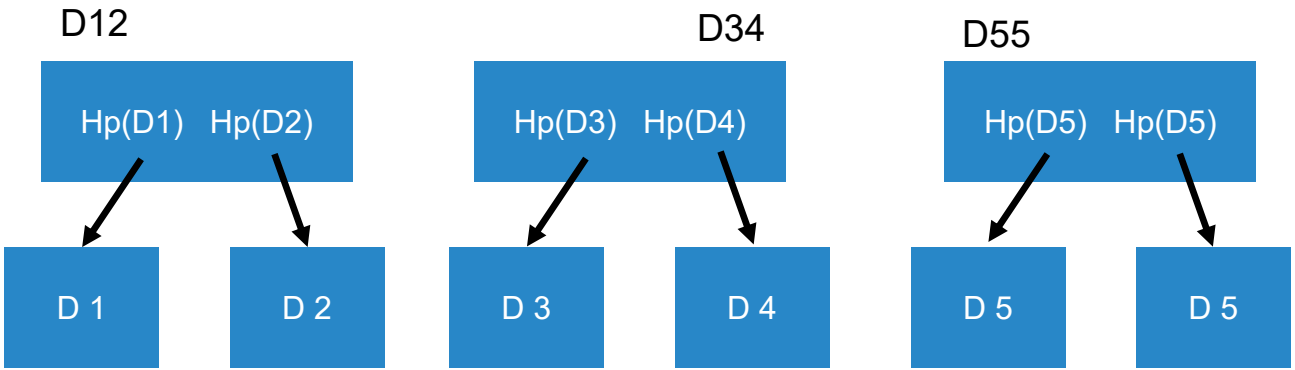
Merkle tree

If I don't have 2^n data?



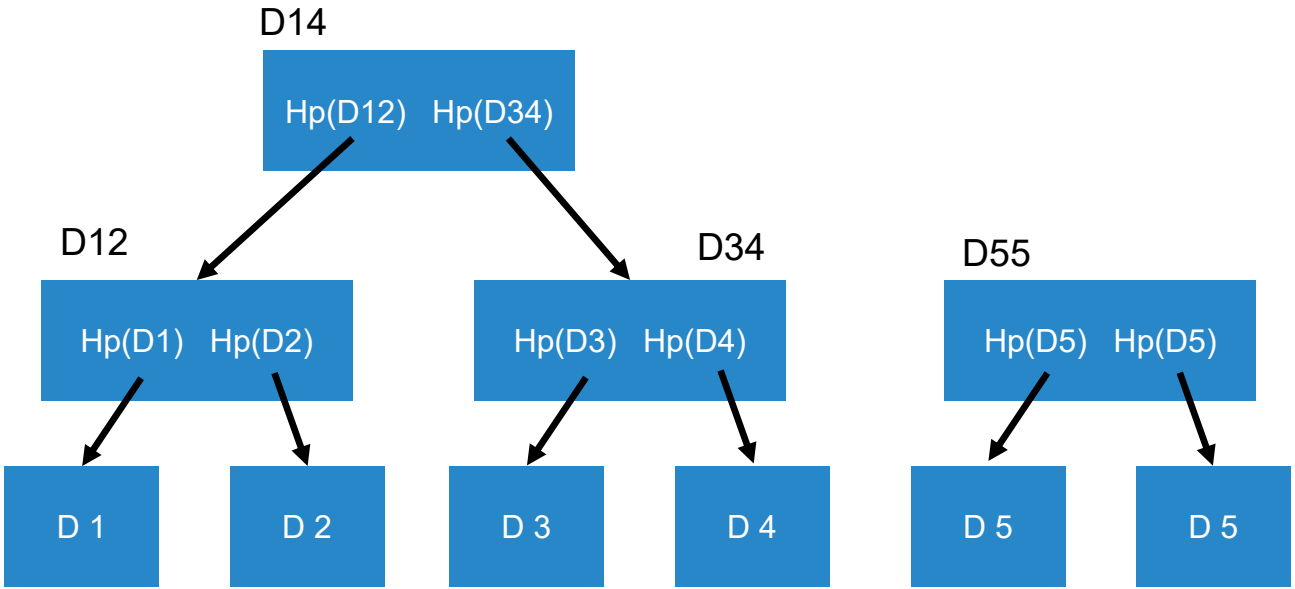
Merkle tree

If I don't have 2^n data?



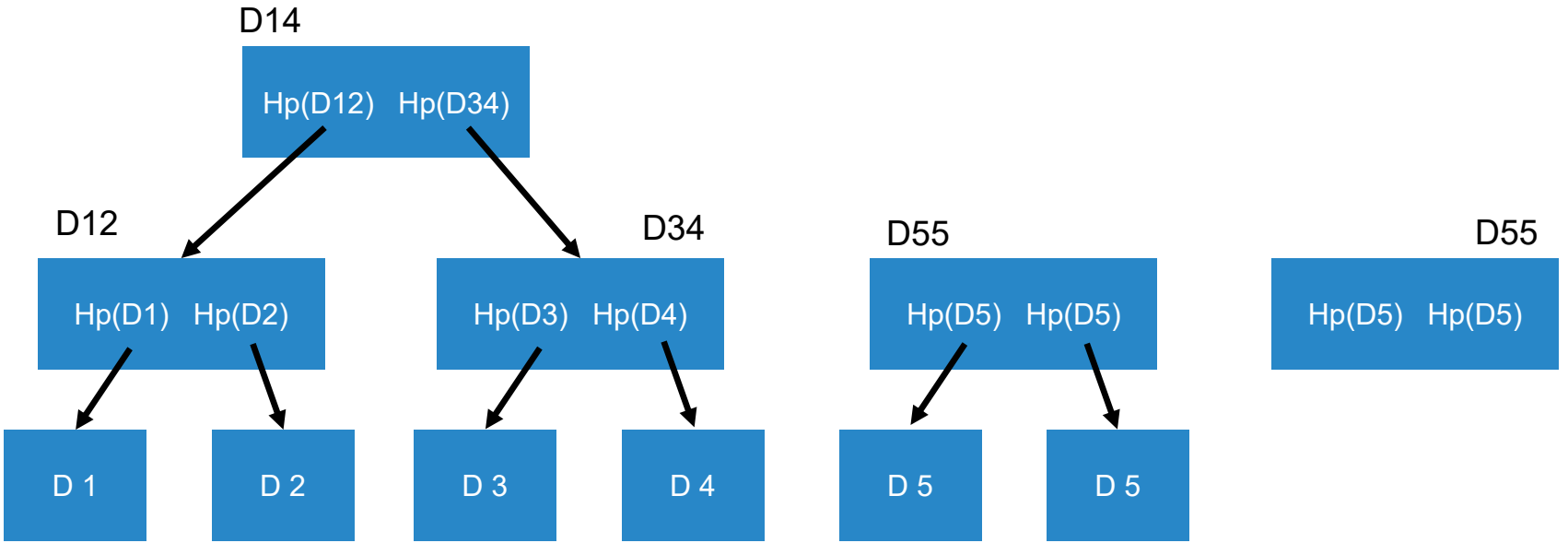
Merkle tree

If I don't have 2^n data?



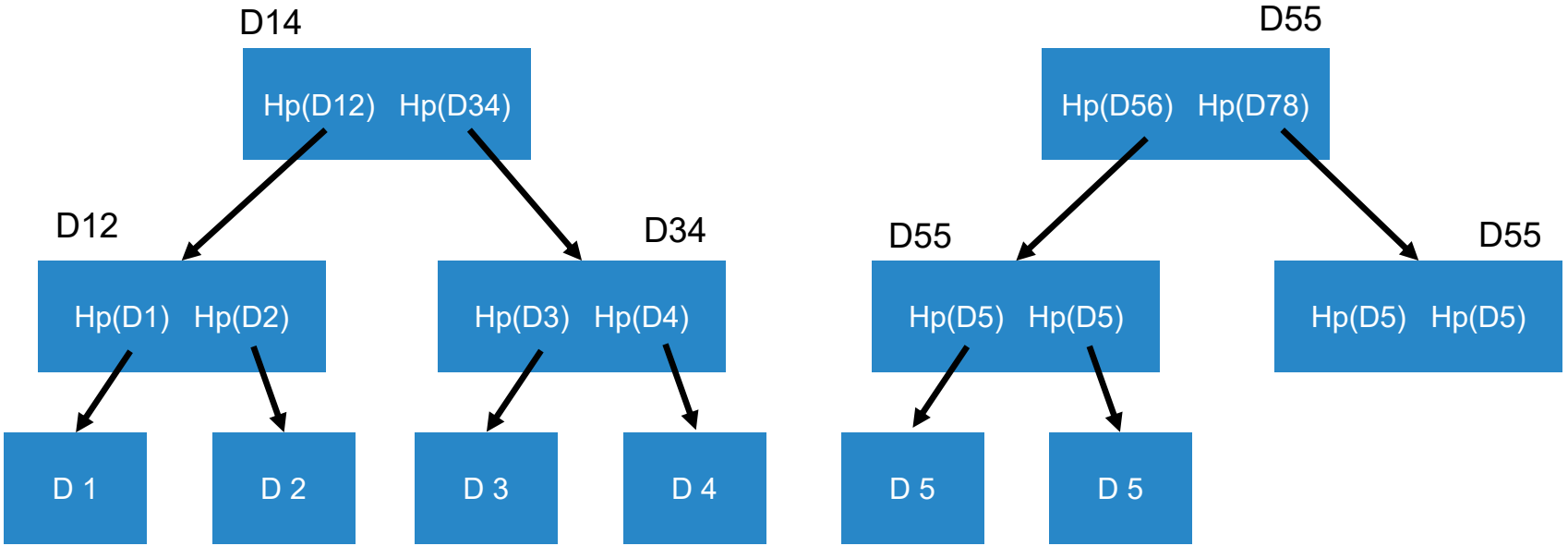
Merkle tree

If I don't have 2^n data?



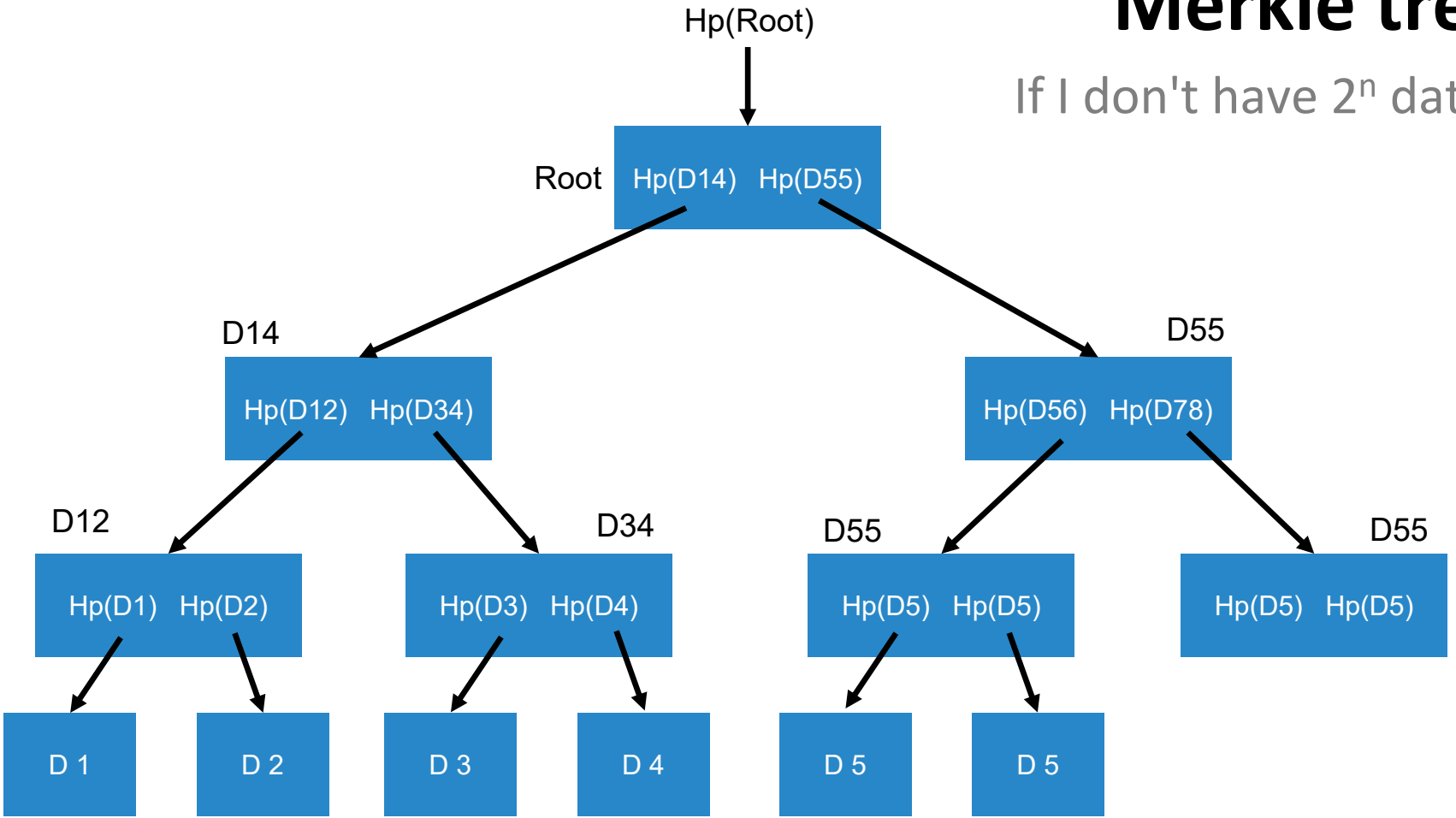
Merkle tree

If I don't have 2^n data?



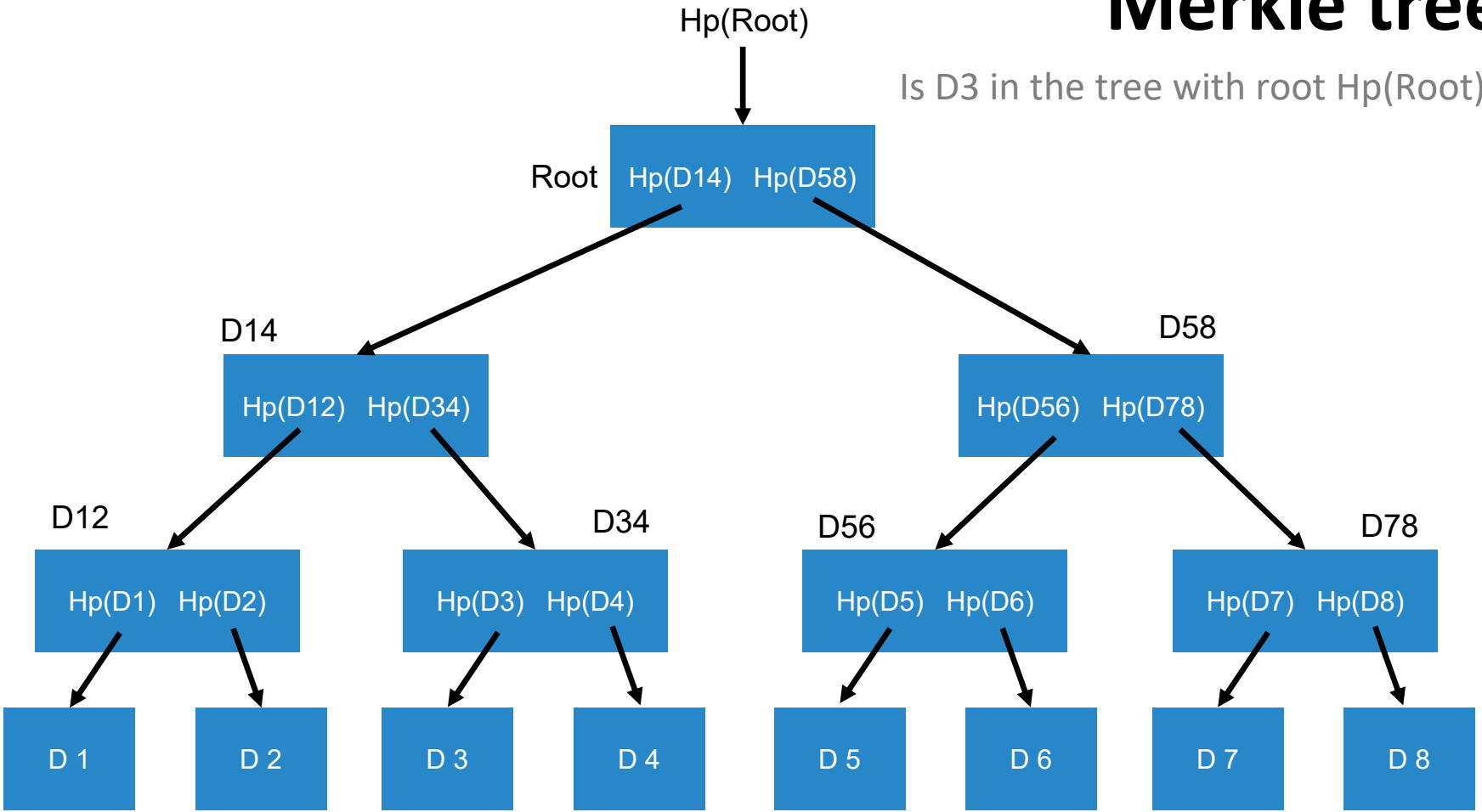
Merkle tree

If I don't have 2^n data?



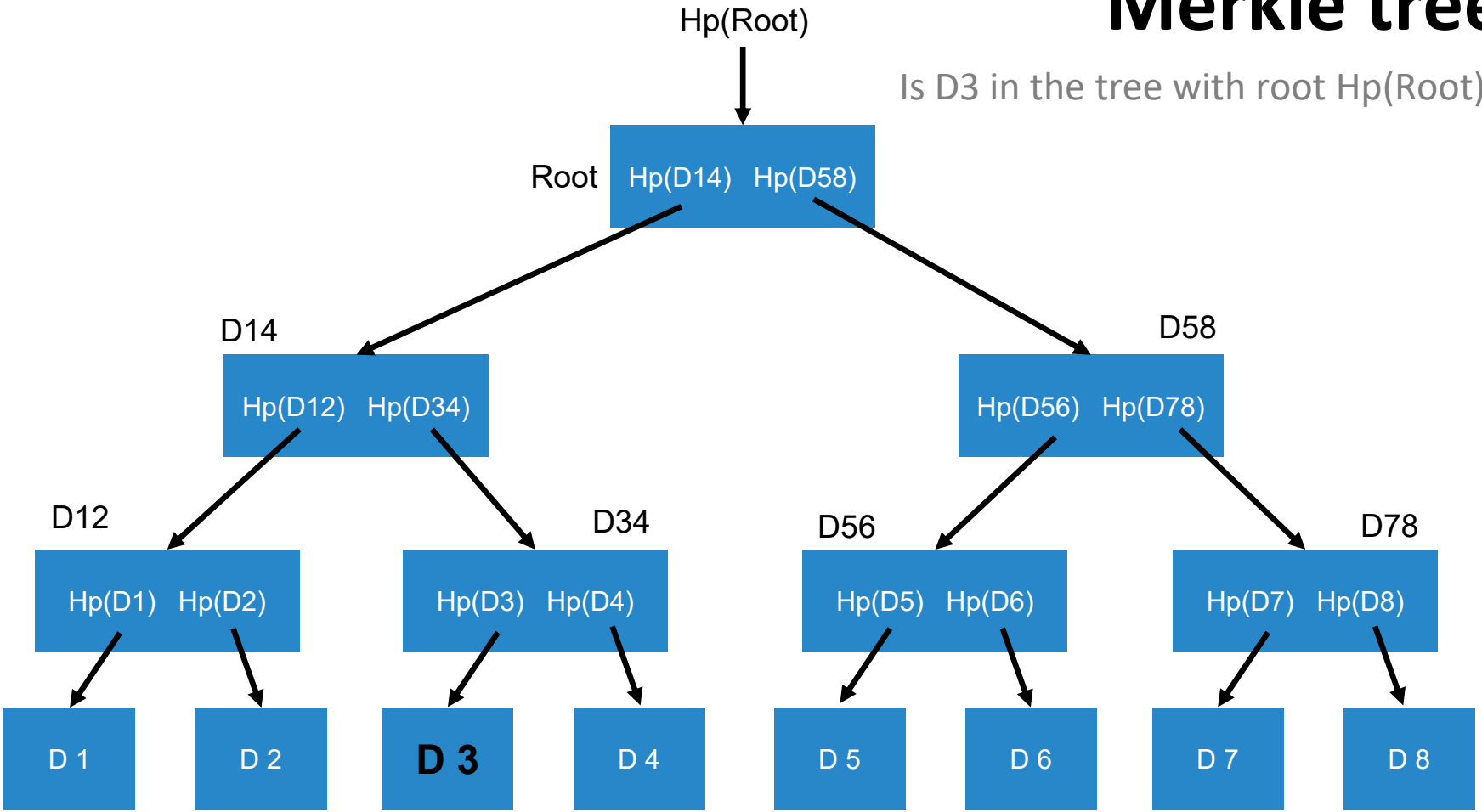
Merkle tree

Is D3 in the tree with root Hp(Root)?



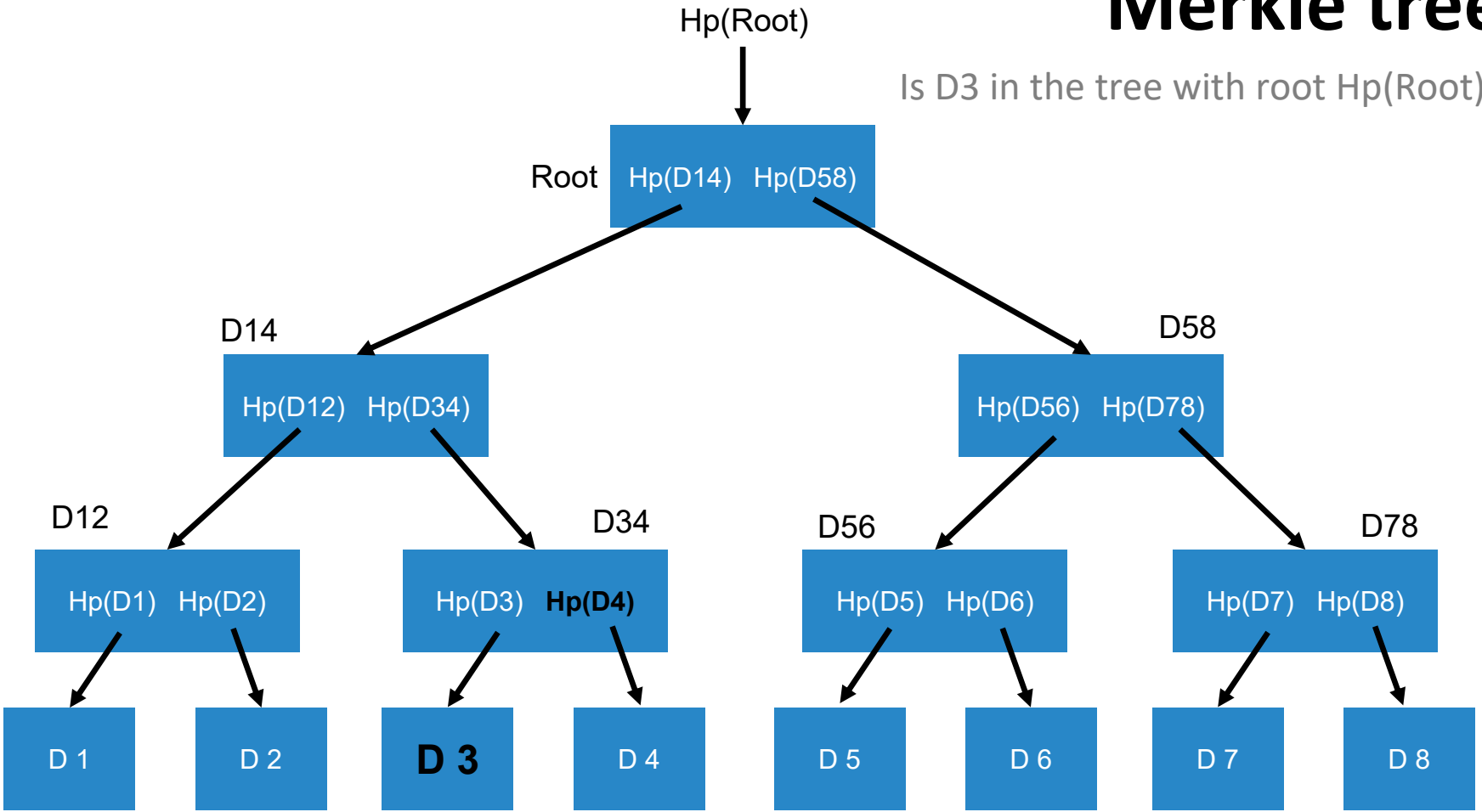
Merkle tree

Is D3 in the tree with root Hp(Root)?



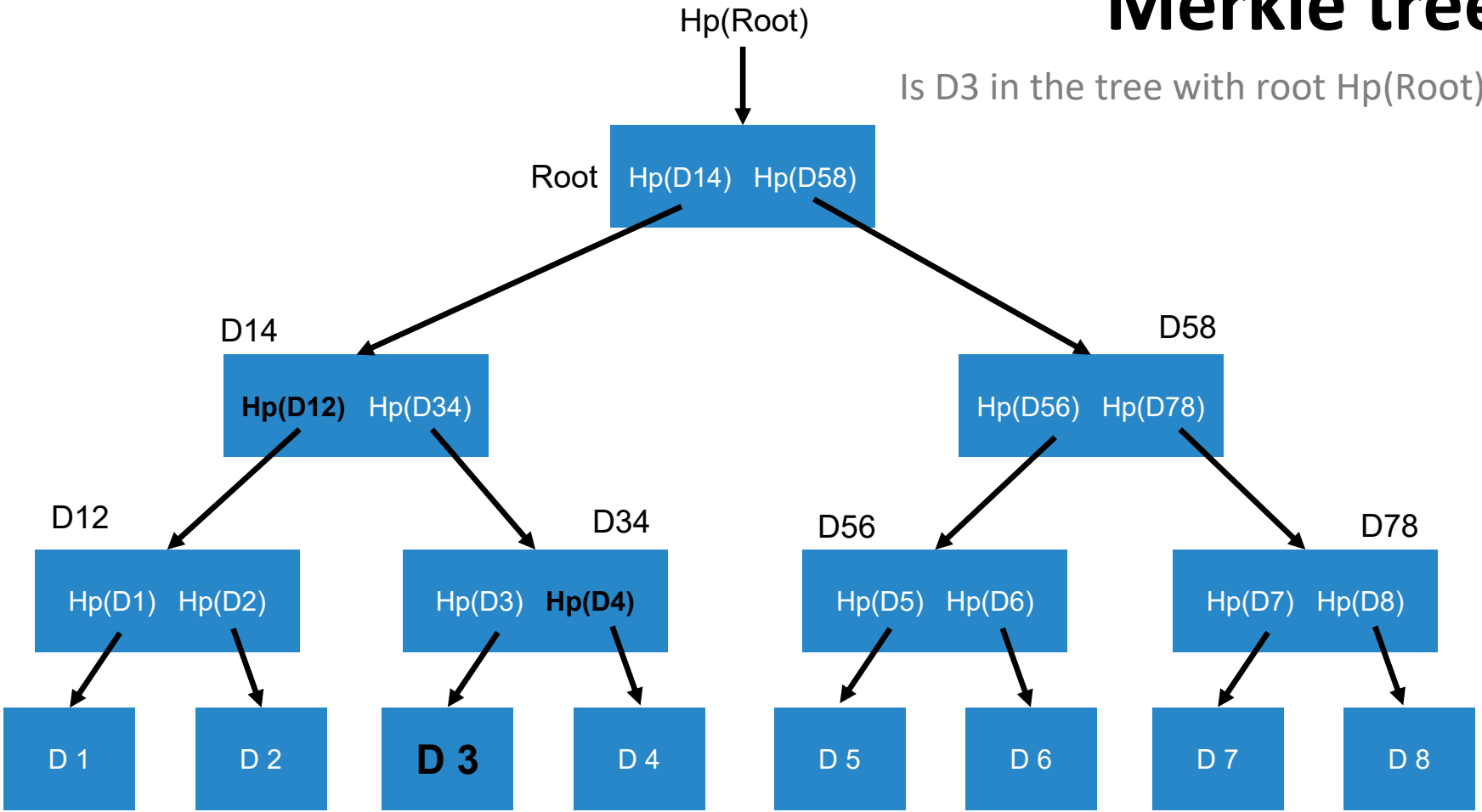
Merkle tree

Is D3 in the tree with root Hp(Root)?



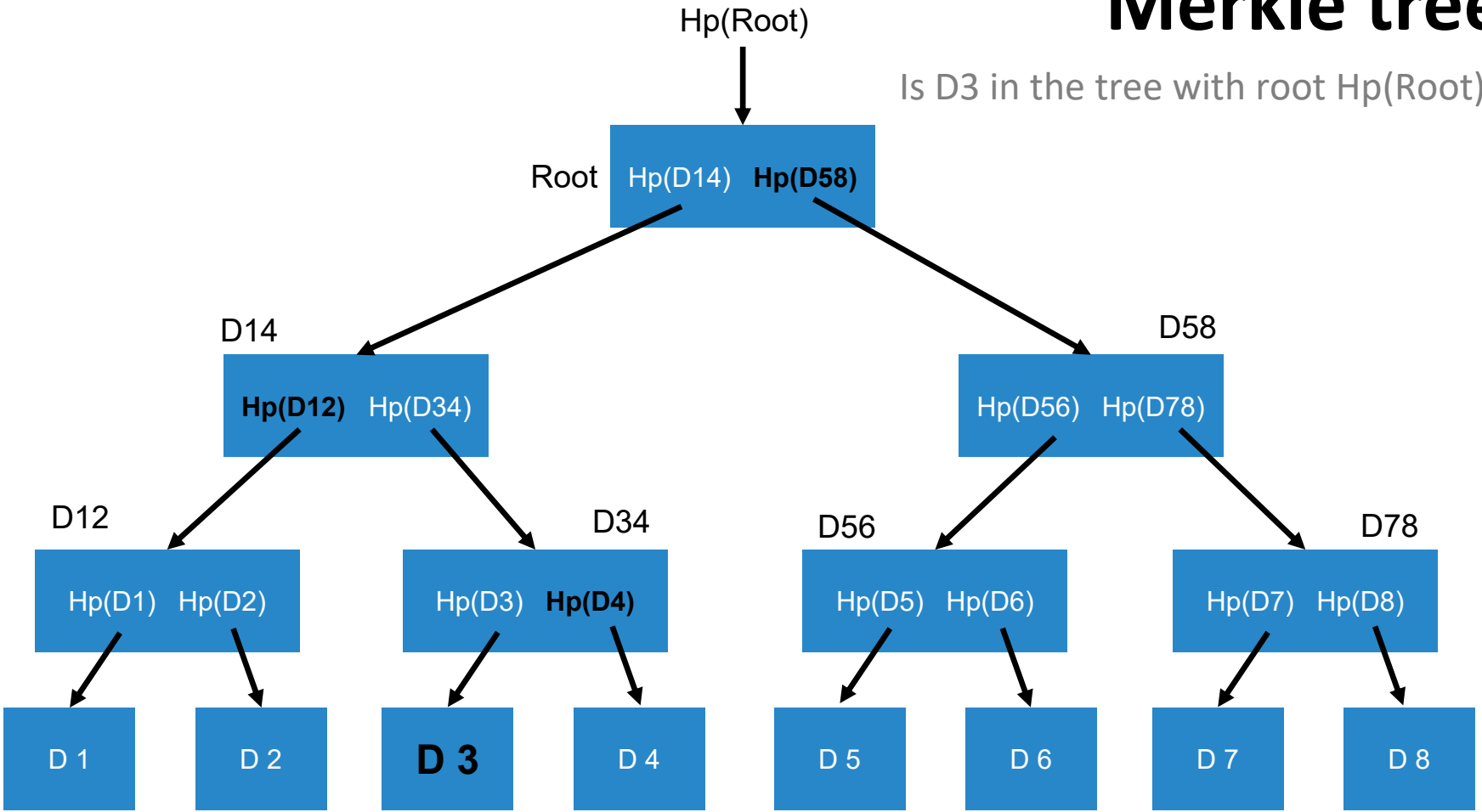
Merkle tree

Is D3 in the tree with root Hp(Root)?



Merkle tree

Is D3 in the tree with root Hp(Root)?



Proof of membership

To show that Di belongs to the tree with the root Hp(Root):

- D_i
- Hp(Neighbours on the path to the root)

Total of $\log_2(n)$ hashes to verify that D_i belongs to the tree

Example: 1024 D_i each of size 1GB = Data Block of 1TB:

- Proof for D_i is of size -- 1GB (D_i) + $\log_2(1024)$ = 10 hashes (2560 bits)
- An order of magnitude less that the size of the Data Block

Proof of non-membership

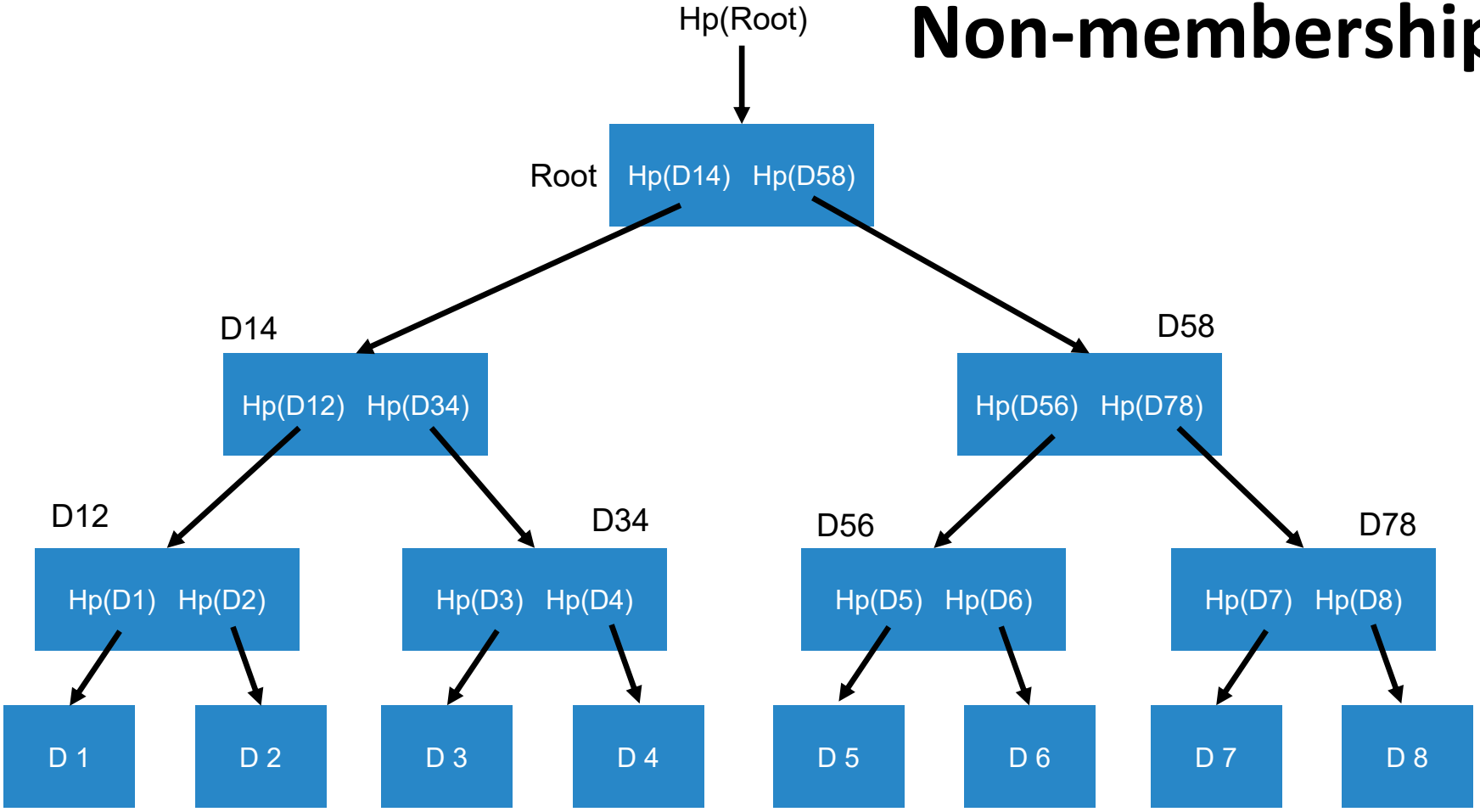
How to show that D ***does not*** belong to the tree with the root Hp(Root)?

Proof of non-membership

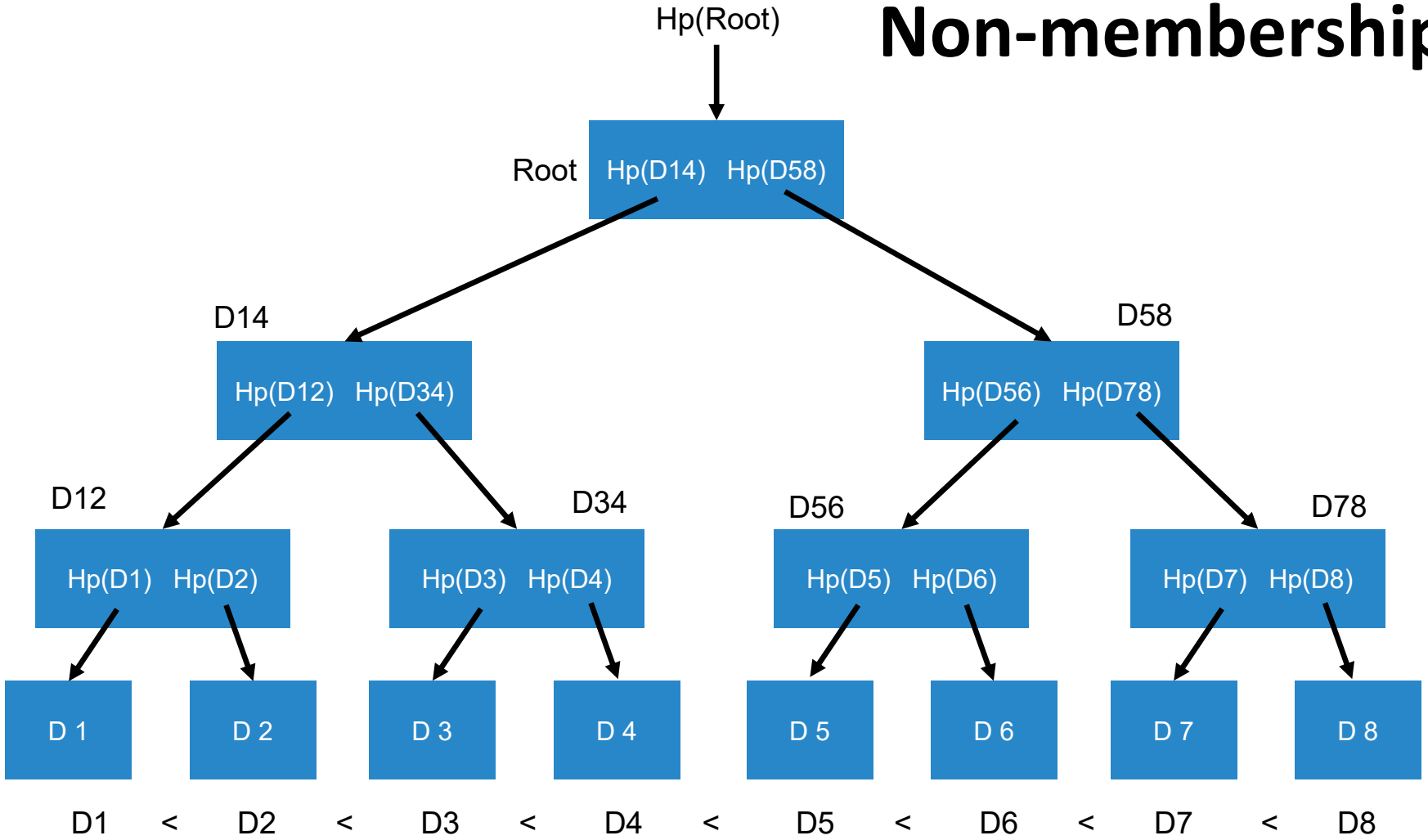
How to show that D ***does not*** belong to the tree with the root Hp(Root)?

Let’s order the data in the leaves first!!!

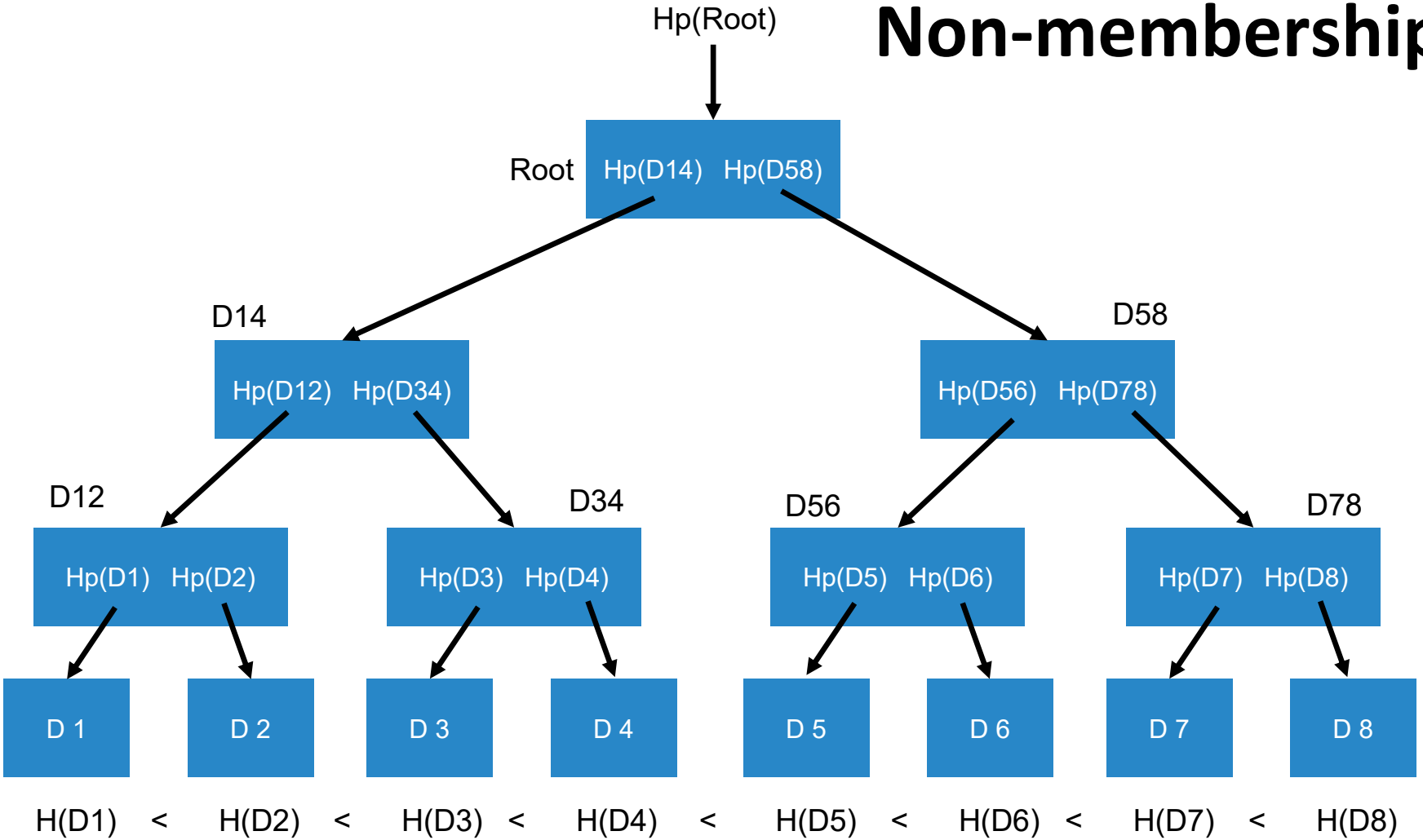
Non-membership



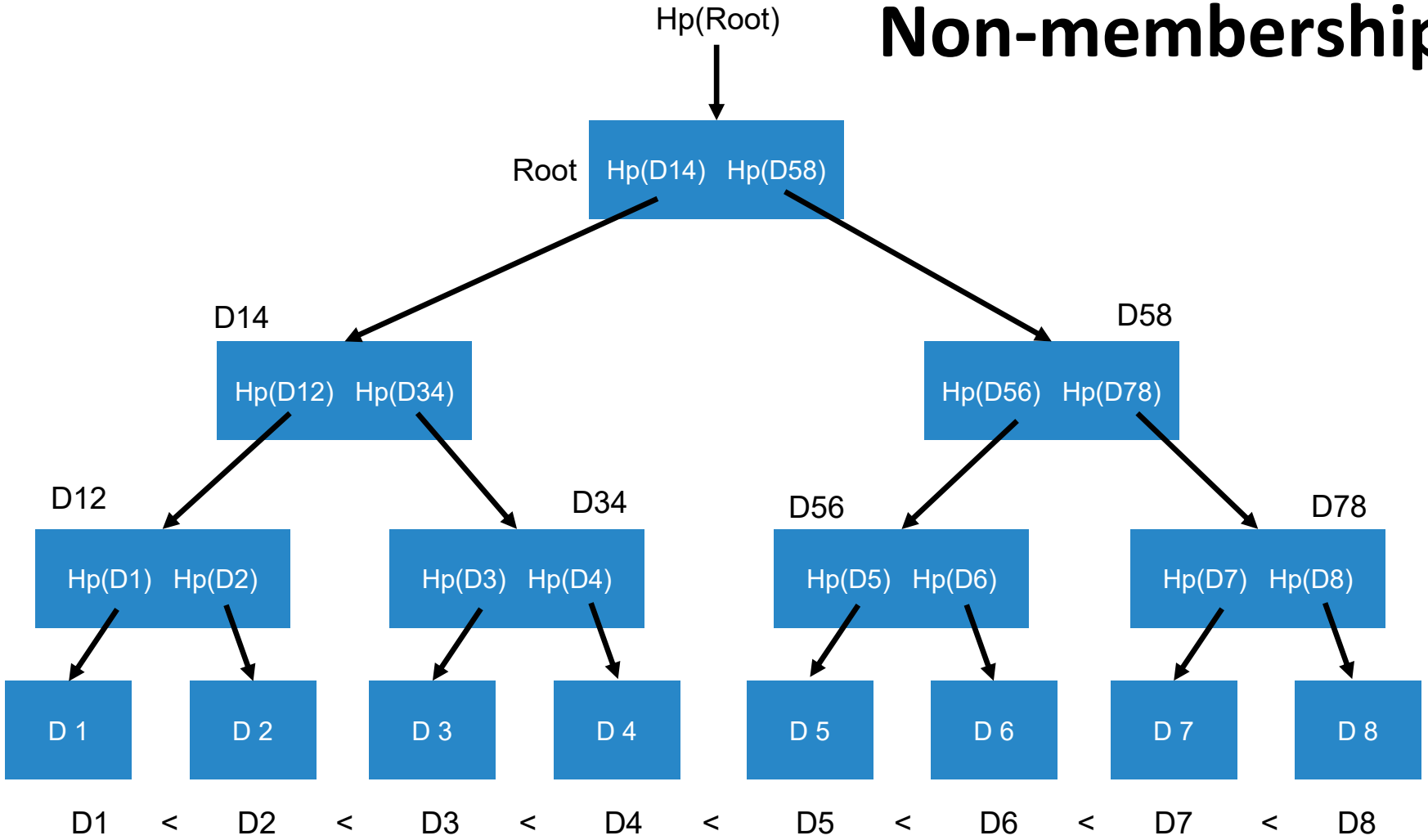
Non-membership



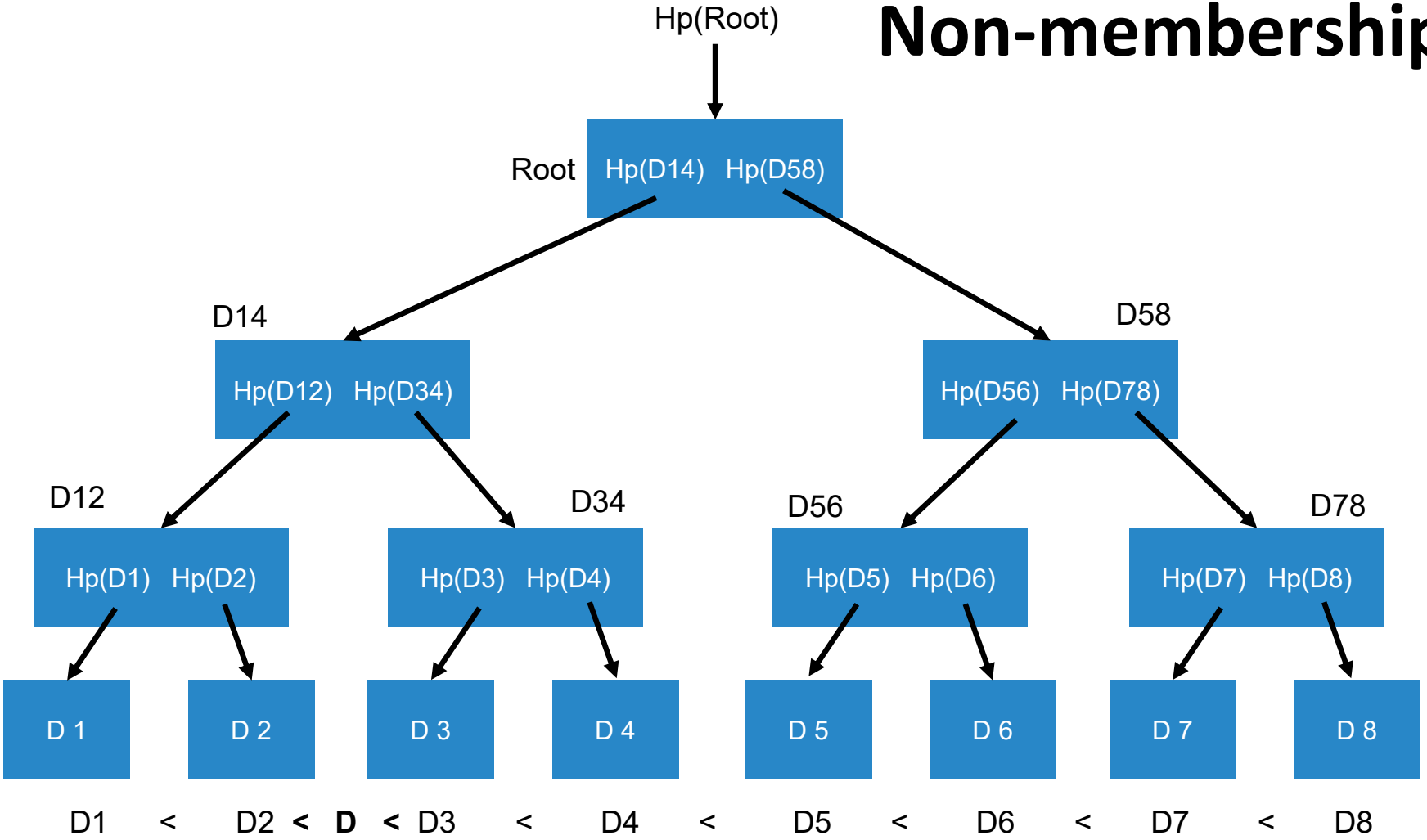
Non-membership



Non-membership



Non-membership



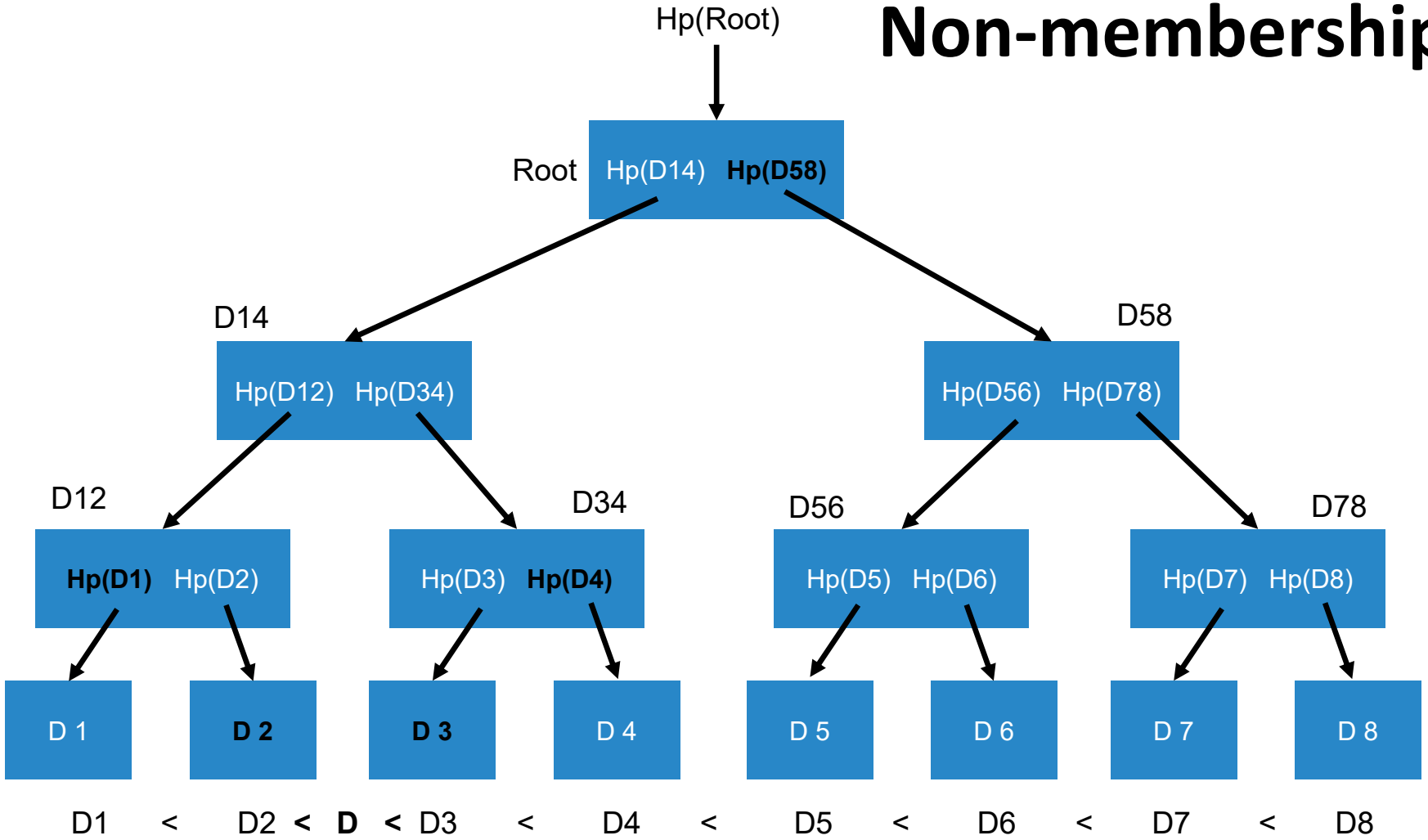
Proof of non-membership

To show that D ***does not*** belong to the tree with root Hp(Root):

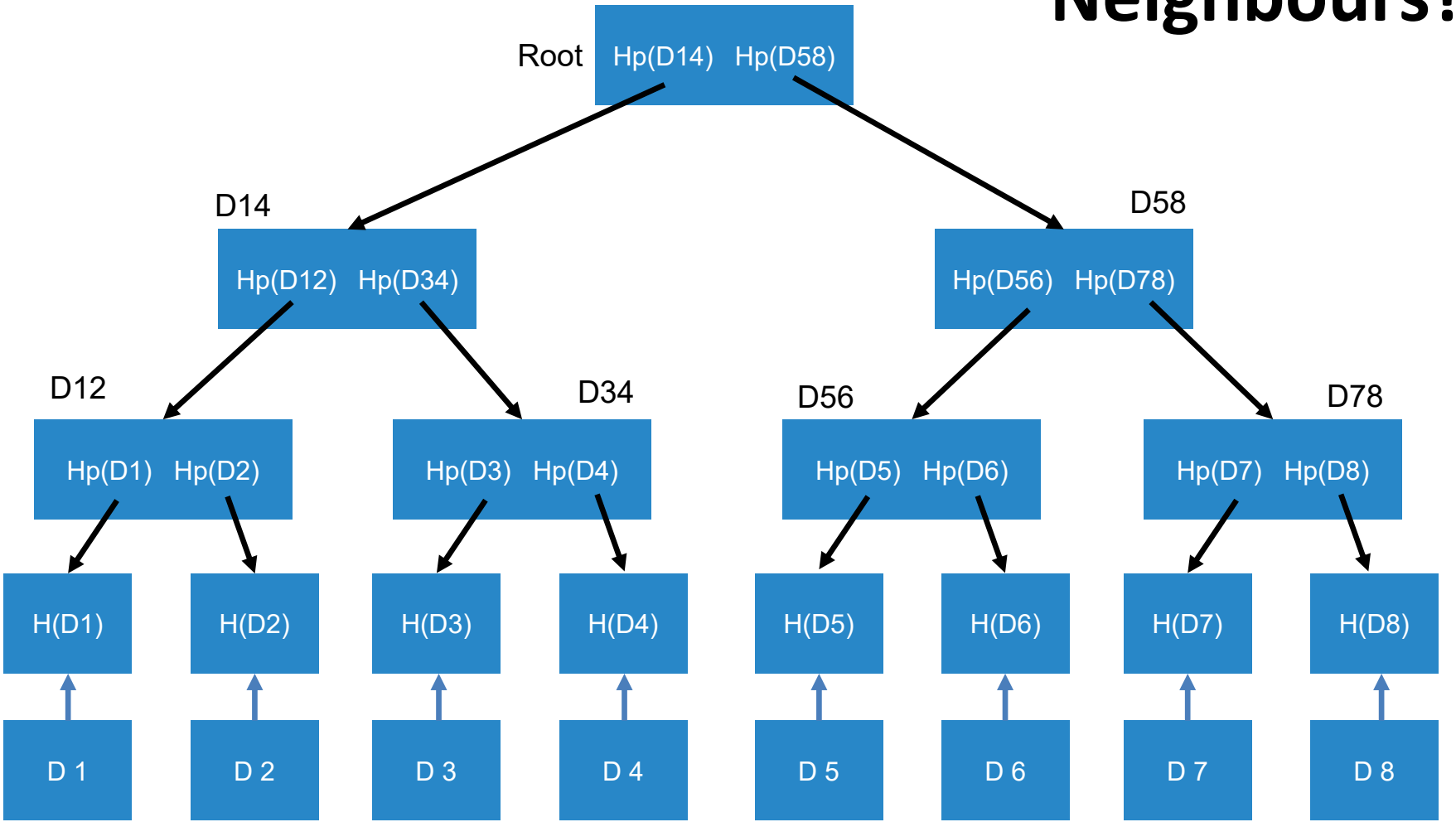
- Minimal *i* s.t. $D_i < D$
- Proof of membership for D_i and D_{i+1}
- Proof that D_i and D_{i+1} are Neighbours in the tree (**how to do this???**)

Since D_i y D_{i+1} are Neighbours in the tree, D does not fit inside

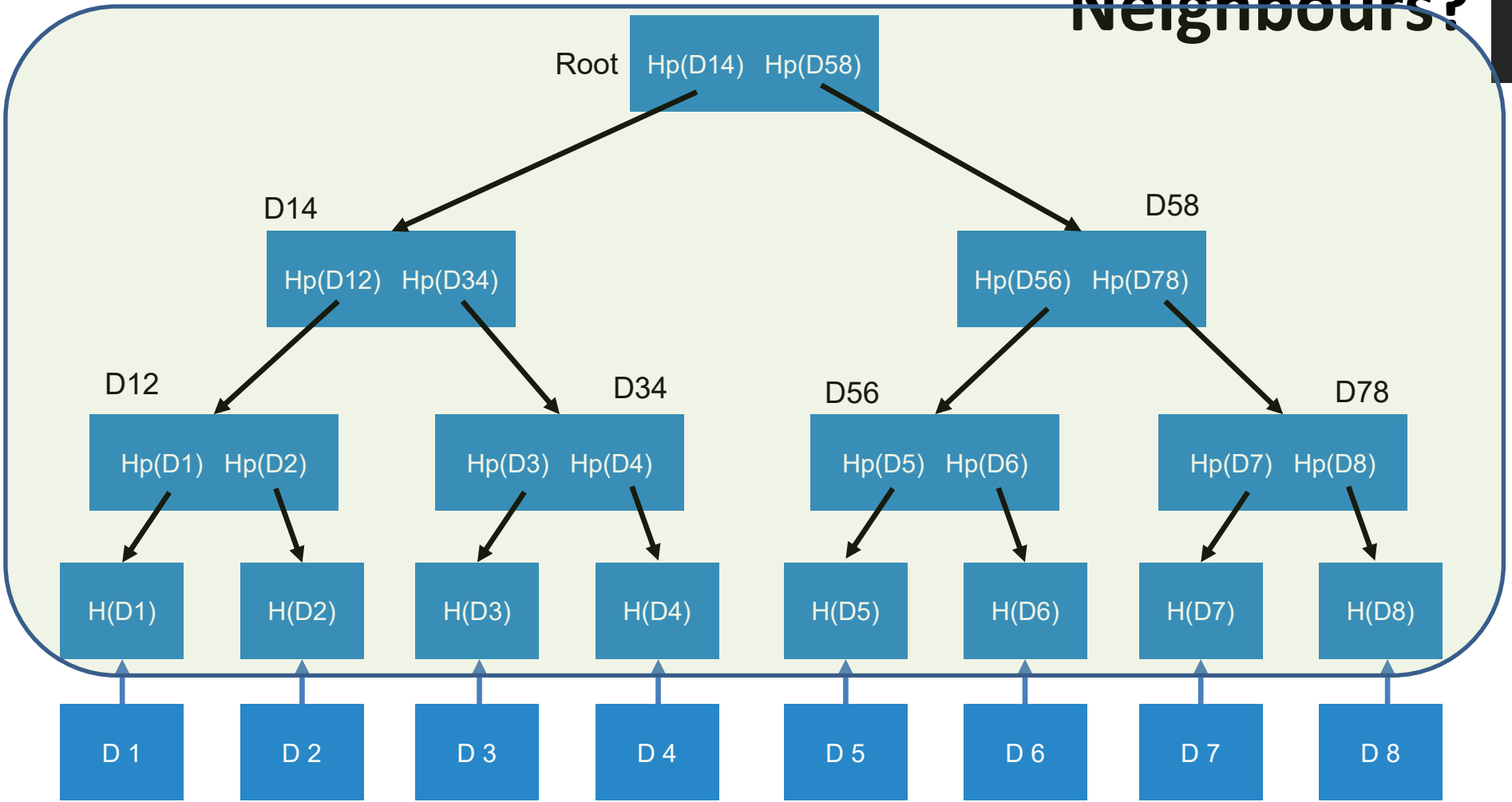
Non-membership



Neighbours?


Root
Hp(D14) Hp(D58)
D14
Hp(D12) Hp(D34)
D58
Hp(D56) Hp(D78)
D12
Hp(D1) Hp(D2)
D34
Hp(D3) Hp(D4)
D56
Hp(D5) Hp(D6)
D78
Hp(D7) Hp(D8)
H(D1)
H(D2)
H(D3)
H(D4)
H(D5)
H(D6)
H(D7)
H(D8)
D 1
D 2
D 3
D 4
D 5
D 6
D 7
D 8

Neighbours?


Root
Hp(D14) Hp(D58)
D14
Hp(D12) Hp(D34)
D58
Hp(D56) Hp(D78)
D12
Hp(D1) Hp(D2)
D34
Hp(D3) Hp(D4)
D56
Hp(D5) Hp(D6)
D78
Hp(D7) Hp(D8)
H(D1)
H(D2)
H(D3)
H(D4)
H(D5)
H(D6)
H(D7)
H(D8)
D 1
D 2
D 3
D 4
D 5
D 6
D 7
D 8

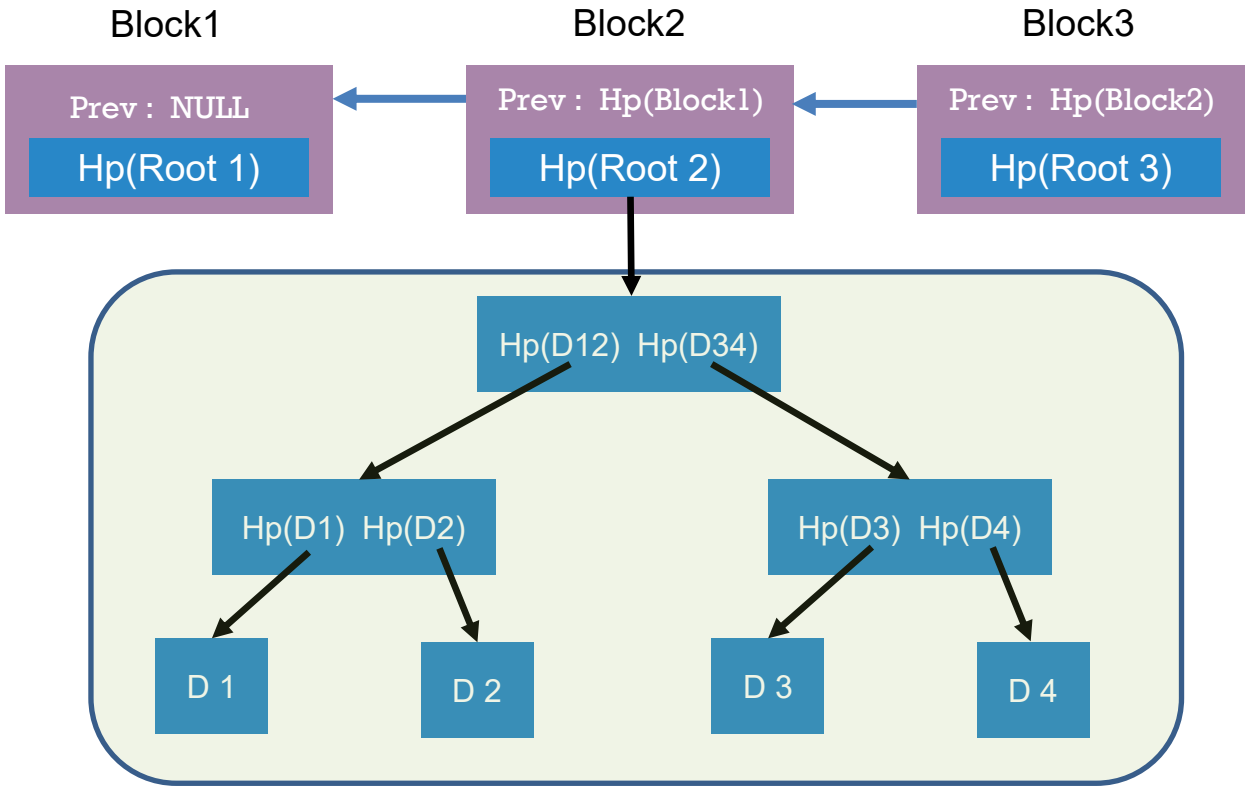
What happens in BitCoin?

Can I use trees to have a tamper-evident log?

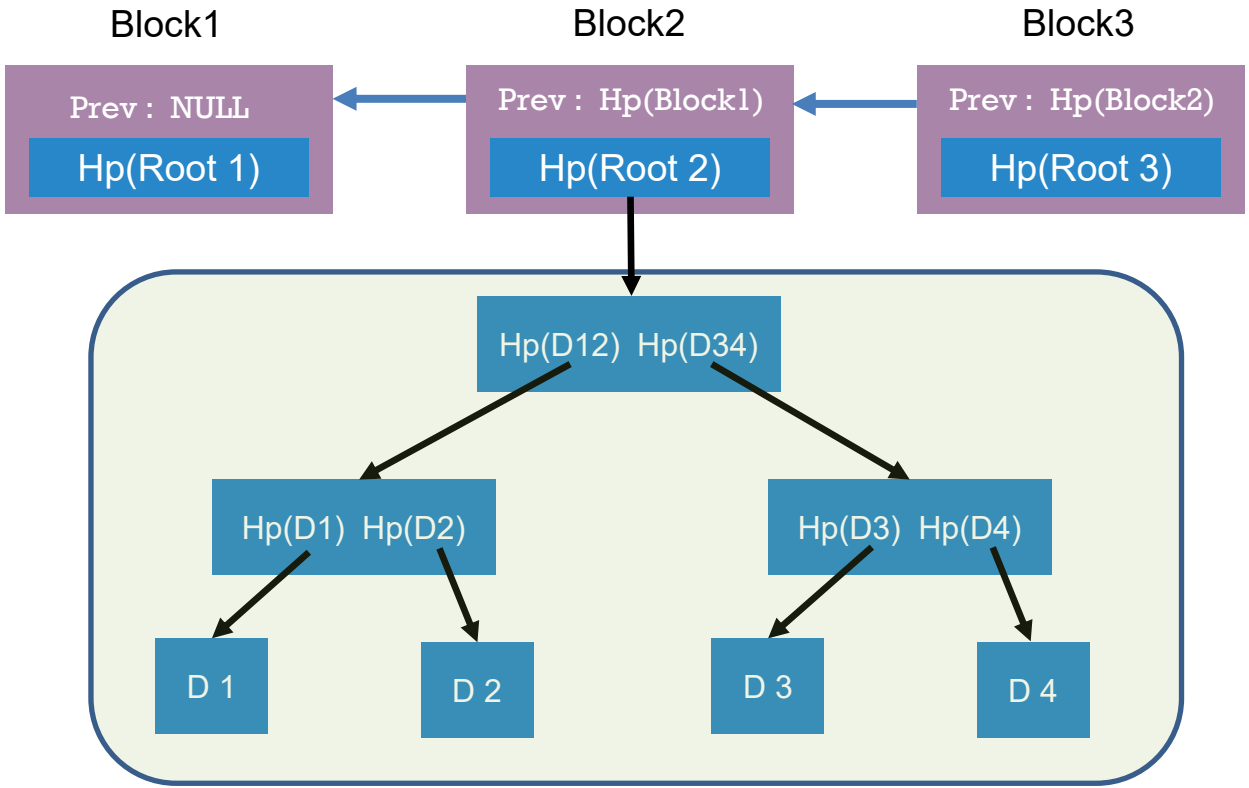
BlockChain:

- Data in a single block = Merkle tree
- Blocks arranged in a blockchain

What happens in BitCoin?



What happens in BitCoin?



In BitCoin:

D_i is a transaction

References

Reading:

- Chapter 1.3 of Narayana et. al.
- Chapter 11 of Programming Bitcoin (Jimmy Song)