Range-Based Set Reconciliation

Aljoscha Meyer

Set Reconciliation

- set union over a network
- between (exactly) two machines
- unstructured data
- no shared state or history

Trivial Reconciliation

Sam Gwilym

Model and Analysis

- Alfie and Betty talk over a network
- reliable communication, rounds of unit length, unlimited bandwidth
- probabilistic solutions

Model and Analysis

- Alfie and Betty talk over a network
- reliable communication, rounds of unit length, unlimited bandwidth
- probabilistic solutions
- n: size of the union
- $n_{\triangle}$: size of the symmetric difference

Model and Analysis

- roundtrips
- communicated bytes
- computation time per reconciliation session
- computation space per reconciliation session
- computation time per item
- computation space per item

P2P Reconciliation

Peer-to-peer systems:

- iterating over local set infeasible
- loading local set into memory infeasible
- some peers are out to get us

P2P Reconciliation

Peer-to-peer systems:

- iterating over local set infeasible
- loading local set into memory infeasible
- some peers are out to get us

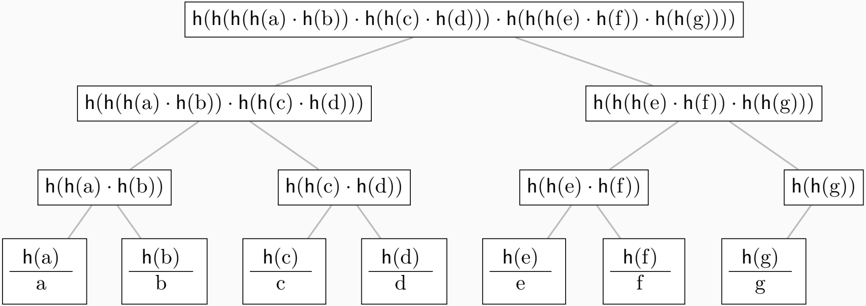
$\implies$ traditional approaches don't work

Reducing Computation Times

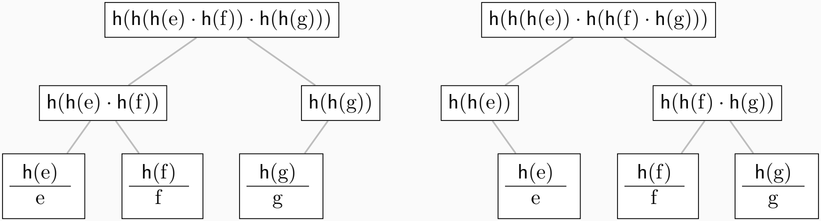
- Step 1: Put a Merkle tree on it
- Step 2: ???
- Step 3: Profit

Auvolat, Alex, and François Taïani. "Merkle search trees: Efficient state-based CRDTs in open networks." 2019 38th Symposium on Reliable Distributed Systems (SRDS). IEEE, 2019.

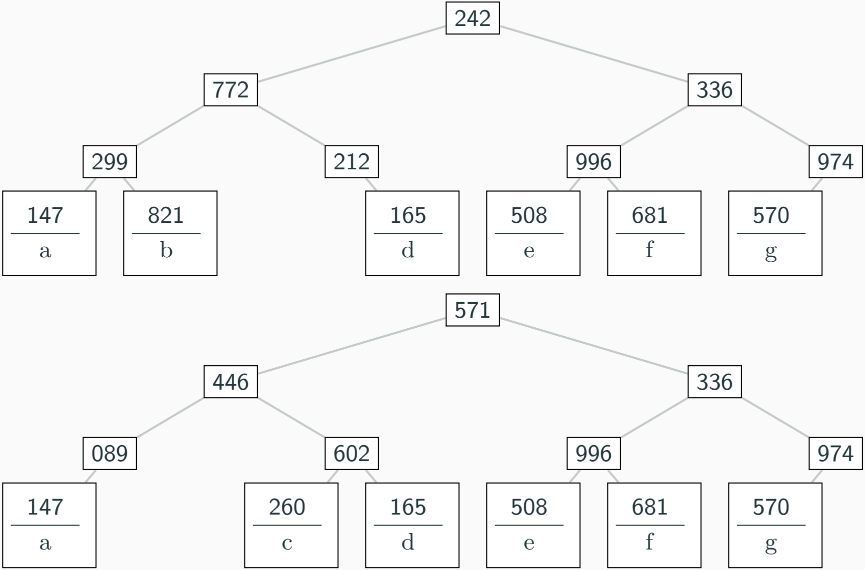
Merkle Trees



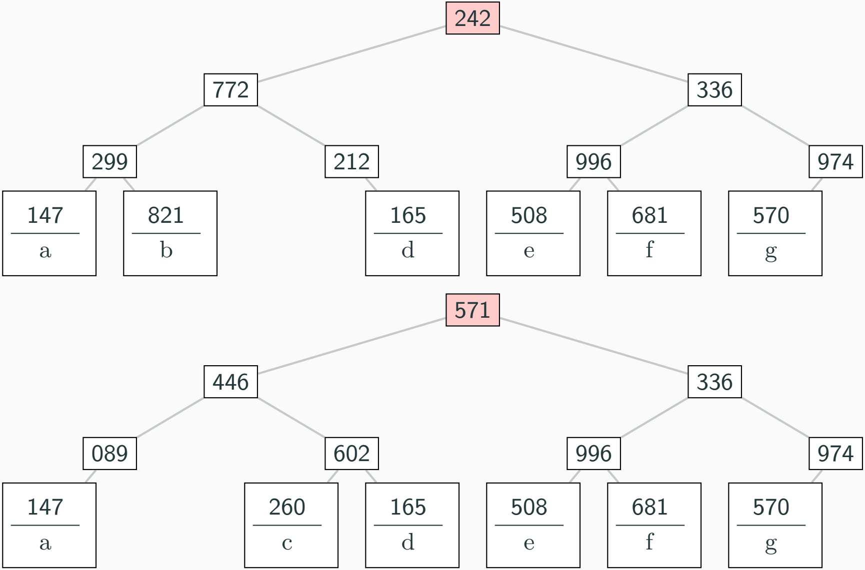
Merkle Trees



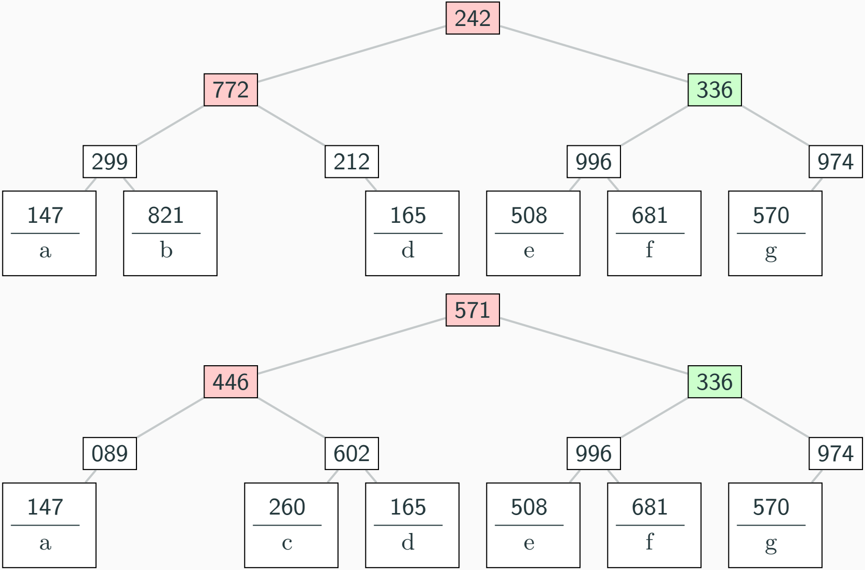
Merkle Tree Reconciliation



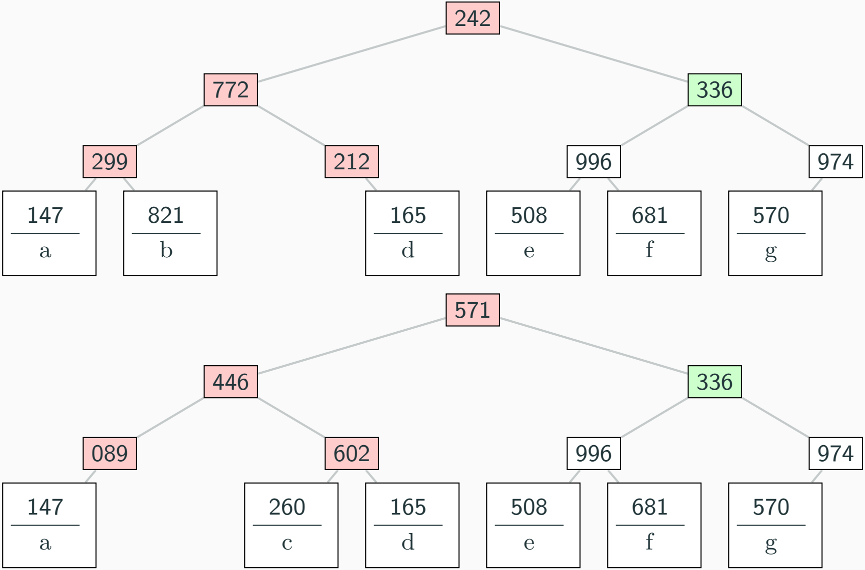
Merkle Tree Reconciliation



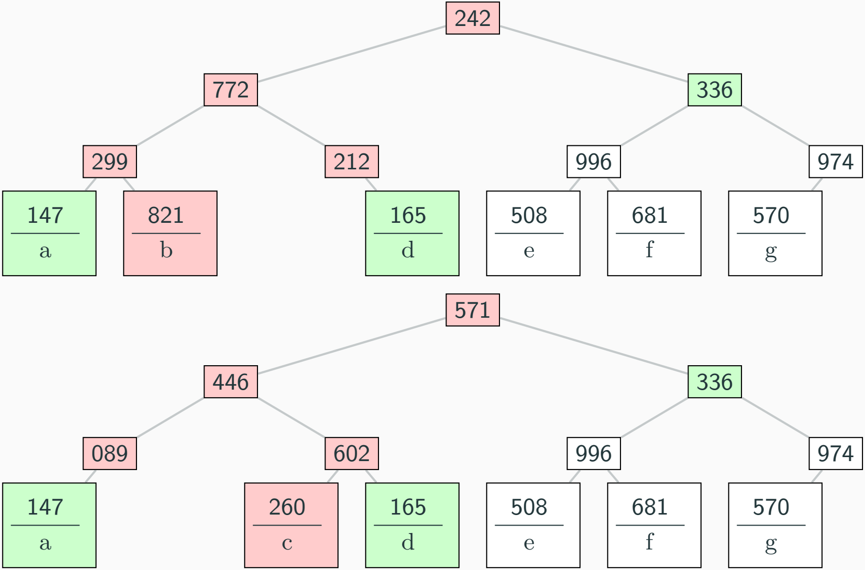
Merkle Tree Reconciliation



Merkle Tree Reconciliation



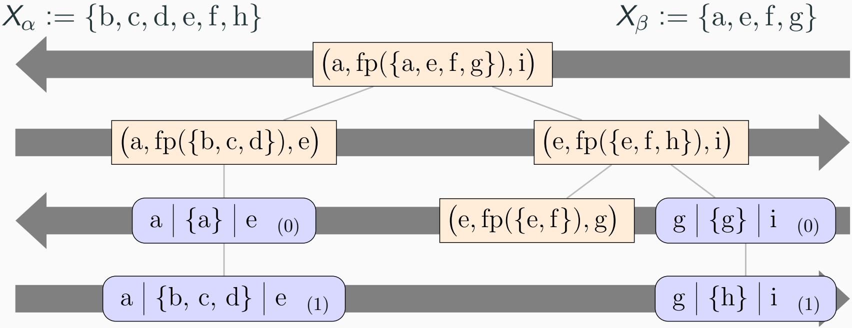
Merkle Tree Reconciliation



Merkle Tree Reconciliation

- inflexible data representation
- inacceptable worst-case complexity
  - remember, some peers are out to get us

Range-Based Set Reconciliation



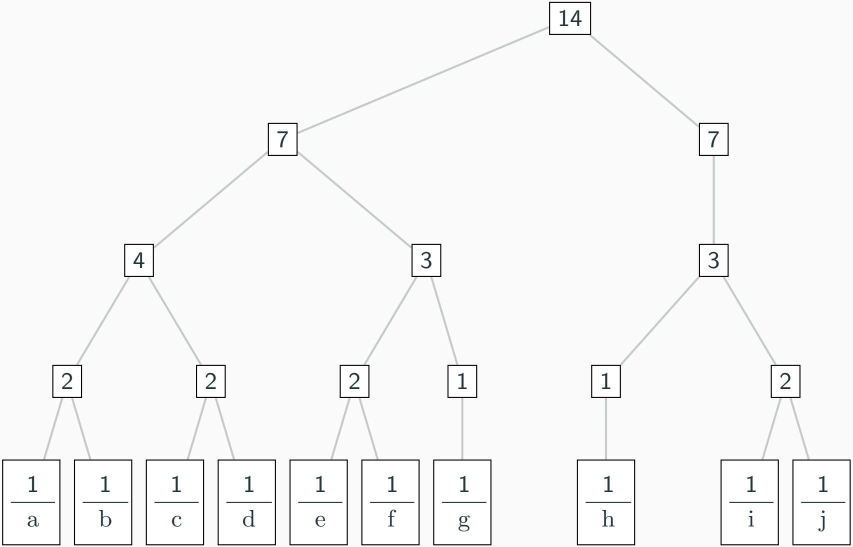
Some Nice Properties

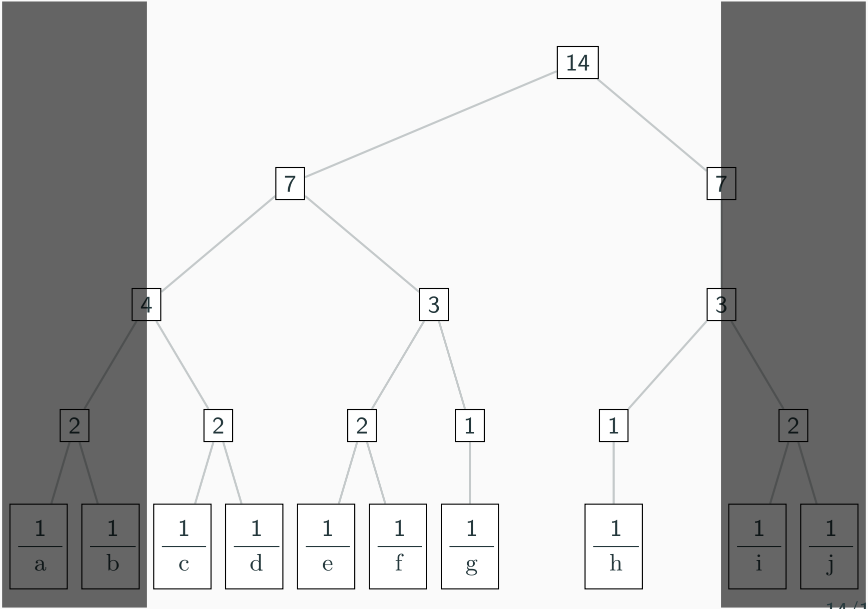
- reasonably efficient: $\mathcal{O}(\min(n_\triangle \cdot \log(n), n))$ bytes communication, $\mathcal{O}(1)$ working memory
- can interpolate toward trivial
- arbitrary recursion anchor protocols
- arbitrary partition techniques

Some Nice Properties

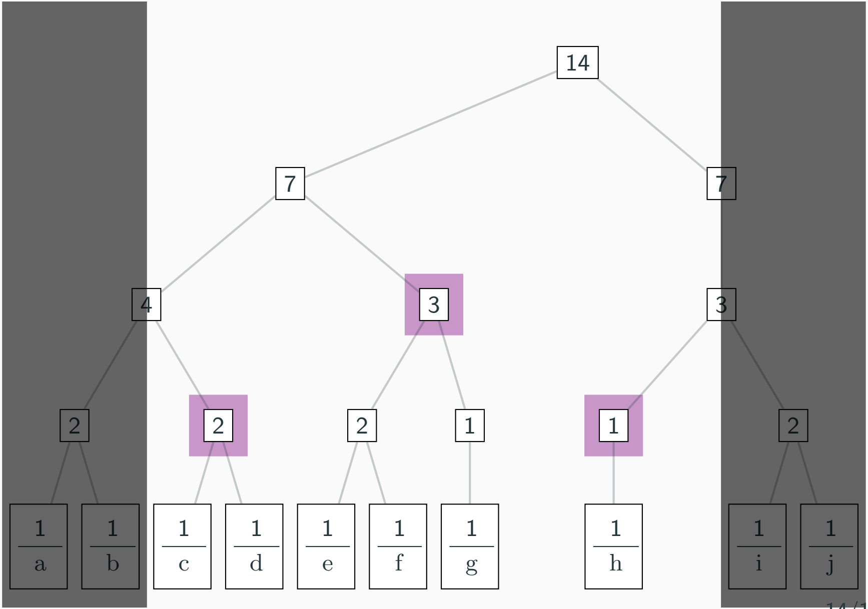
- reasonably efficient: $\mathcal{O}(\min(n_\triangle \cdot \log(n), n))$ bytes communication, $\mathcal{O}(1)$ working memory
- can interpolate toward trivial
- arbitrary recursion anchor protocols
- arbitrary partition techniques
- but: linear computation times

Order-Statistic Trees

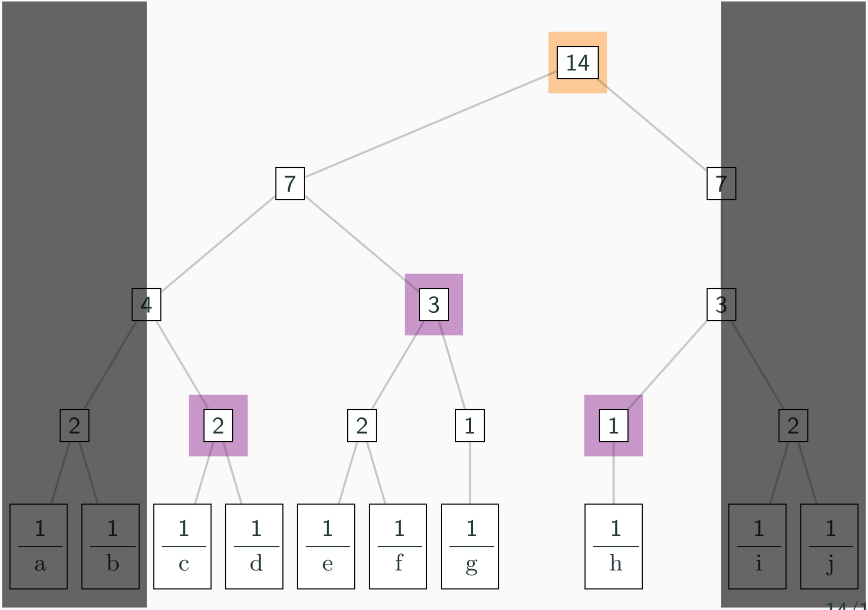



14
7
7
4
3
3
2
2
2
1
1
2
1
a
1
b
1
c
1
d
1
e
1
f
1
g
1
h
1
i
1
j

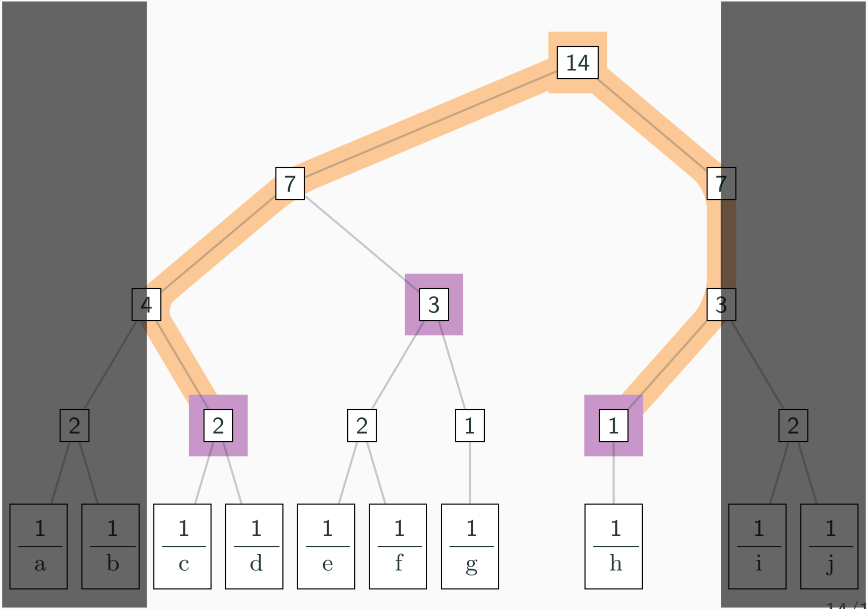
Order-Statistic Trees $[\mathrm{c}, \mathrm{i})$



Order-Statistic Trees $[c, i)$



Order-Statistic Trees $[c, i)$



Monoid Trees

- set of labels: $\mathbb{N}$
- binary associative function: $+$
- neutral element: 0

Monoid Trees

- set of labels: $\mathbb{N}$
- binary associative function: $+$
- neutral element: 0
- lifting into the monoid: $\lambda x.1$

- let's backtrack and solve the first problem -¿ associativity
- example tree (using counts, then hashes)
- definition
- arbitrary ranges: collections of maximal subtrees, complexity analysis
- successive ranges yay

- this gives us worst-case bounds on roundtrips
- implementation independence
- more datastructure-neutral formulation: monoids
- paper actually uses different data structure, also B-trees, skip lists, radix trees, no data structure, ... and all interoperable

- malicious adversaries: active vs passive
- even passive finds collisions in relatively small sets

- but: collision must actually affect a session -¿ brittle, and randomization ftw
- But what if there are many collisions? -¿ collision-resistant hash functions

- must always lift into the monoid with a secure hash function. Interesting part is the selection of the monoid
- bellare: xor, addition, multiplication, vectorized addition (lattices)
- multiset homomorphic hashing adds rsa and elliptic curves
- we are weaker: homomorphism-flavored characterization
- in particular, no need for commutativity (because we use search trees) -¿ matrix multiplication -¿ cayley hashes
- Don't know anything else, but cayley still has unnecesary structure, as we do not need inverses

- conclusion: first non-trivial reconciliation algorithm to work in resource-constrained settings and adversarial environments
- also: comparatively simple -¿ had multiple open source developers reach out to me