# Efficient Synchronization of Recursively Partitionable Data Structures

# Synchronizing Data Structures

- What?
  - set reconciliation (aka set union)
  - set mirroring
  - map mirroring
- Why?
  - distributed version control
  - persistent unordered PubSub
  - backup creation
  - update distribution
  - ...

# Evaluation Criteria

- number of round-trips
- total bandwidth usage
- computation time complexity per round-trip
- space complexity per round-trip
- space complexity for auxiliary data structures
- time complexity for keeping auxiliary data structure up to date as the main data structure is being modified
- soft criteria: simplicity and flexibility

# Basic Ideas

**Divide and conquer**: Let $X_0, X_1$ be sets with $X_0 = A_0 \dot\cup B_0$ and $X_1 = A_1 \dot\cup B_1$, then $X_0 \cup X_1 = (A_0 \cup B_0) \cup (A_1 \cup B_1)$.

# Basic Ideas

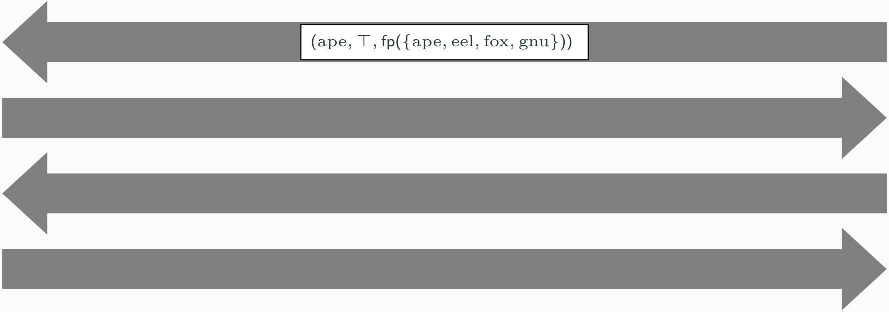
**Divide and conquer**: Let $X_0, X_1$ be sets with $X_0 = A_0 \dot{\cup} B_0$ and $X_1 = A_1 \dot{\cup} B_1$, then $X_0 \cup X_1 = (A_0 \cup B_0) \cup (A_1 \cup B_1)$.

**Fingerprints**: Map sets into $\{0, 1\}^*$ with low probability of collisions.

# Example Protocol Run

$$\mathrm{ape} < \mathrm{bee} < \mathrm{cat} < \mathrm{doe} < \mathrm{eel} < \mathrm{fox} < \mathrm{gnu} < \mathrm{hog} < \top$$

$X_0 := \{\mathrm{bee}, \mathrm{cat}, \mathrm{doe}, \mathrm{eel}, \mathrm{fox}, \mathrm{hog}\}$ $\qquad$ $X_1 := \{\mathrm{ape}, \mathrm{eel}, \mathrm{fox}, \mathrm{gnu}\}$

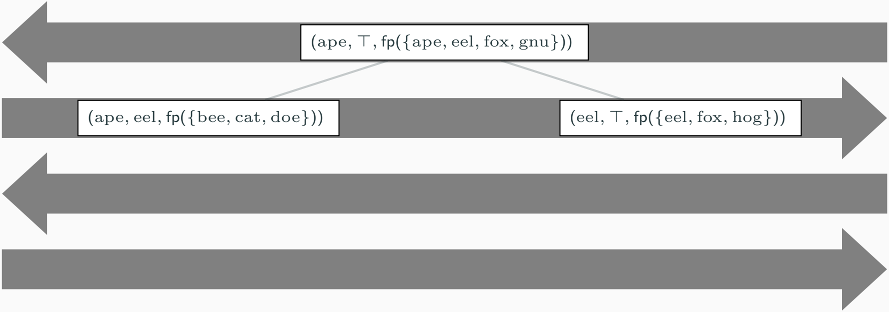


$(\mathrm{ape}, \top, \mathsf{fp}(\{\mathrm{ape}, \mathrm{eel}, \mathrm{fox}, \mathrm{gnu}\}))$

# Example Protocol Run

$$\text{ape} < \text{bee} < \text{cat} < \text{doe} < \text{eel} < \text{fox} < \text{gnu} < \text{hog} < \top$$

$X_0 := \{\text{bee}, \text{cat}, \text{doe}, \text{eel}, \text{fox}, \text{hog}\}$ $\qquad$ $X_1 := \{\text{ape}, \text{eel}, \text{fox}, \text{gnu}\}$



$(\text{ape}, \top, \mathsf{fp}(\{\text{ape}, \text{eel}, \text{fox}, \text{gnu}\}))$

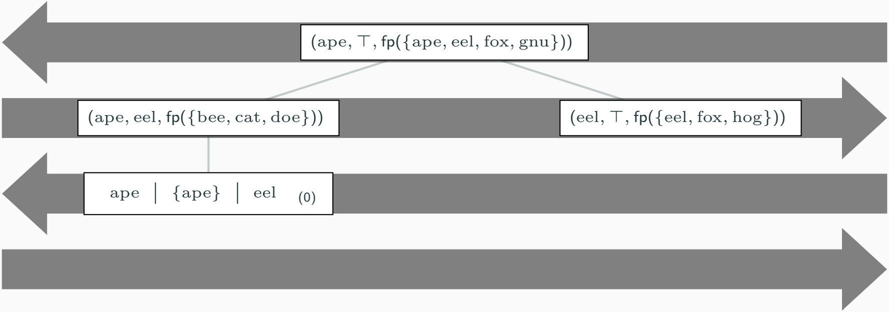
$(\text{ape}, \text{eel}, \mathsf{fp}(\{\text{bee}, \text{cat}, \text{doe}\}))$

$(\text{eel}, \top, \mathsf{fp}(\{\text{eel}, \text{fox}, \text{hog}\}))$

# Example Protocol Run

$$\mathrm{ape} < \mathrm{bee} < \mathrm{cat} < \mathrm{doe} < \mathrm{eel} < \mathrm{fox} < \mathrm{gnu} < \mathrm{hog} < \top$$

$X_0 := \{\mathrm{bee}, \mathrm{cat}, \mathrm{doe}, \mathrm{eel}, \mathrm{fox}, \mathrm{hog}\}$ $X_1 := \{\mathrm{ape}, \mathrm{eel}, \mathrm{fox}, \mathrm{gnu}\}$



$(\mathrm{ape}, \top, \mathsf{fp}(\{\mathrm{ape}, \mathrm{eel}, \mathrm{fox}, \mathrm{gnu}\}))$

$(\mathrm{ape}, \mathrm{eel}, \mathsf{fp}(\{\mathrm{bee}, \mathrm{cat}, \mathrm{doe}\}))$

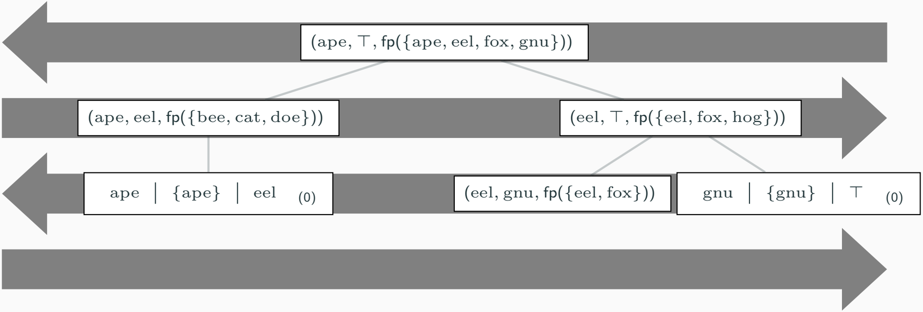
$(\mathrm{eel}, \top, \mathsf{fp}(\{\mathrm{eel}, \mathrm{fox}, \mathrm{hog}\}))$

$\mathrm{ape} \mid \{\mathrm{ape}\} \mid \mathrm{eel}$ (0)

# Example Protocol Run

$$\mathrm{ape} < \mathrm{bee} < \mathrm{cat} < \mathrm{doe} < \mathrm{eel} < \mathrm{fox} < \mathrm{gnu} < \mathrm{hog} < \top$$

$X_0 := \{\mathrm{bee}, \mathrm{cat}, \mathrm{doe}, \mathrm{eel}, \mathrm{fox}, \mathrm{hog}\}$ $X_1 := \{\mathrm{ape}, \mathrm{eel}, \mathrm{fox}, \mathrm{gnu}\}$



$(\mathrm{ape}, \top, \mathsf{fp}(\{\mathrm{ape}, \mathrm{eel}, \mathrm{fox}, \mathrm{gnu}\}))$

$(\mathrm{ape}, \mathrm{eel}, \mathsf{fp}(\{\mathrm{bee}, \mathrm{cat}, \mathrm{doe}\}))$

$(\mathrm{eel}, \top, \mathsf{fp}(\{\mathrm{eel}, \mathrm{fox}, \mathrm{hog}\}))$

$\mathrm{ape} \mid \{\mathrm{ape}\} \mid \mathrm{eel}$ (0)

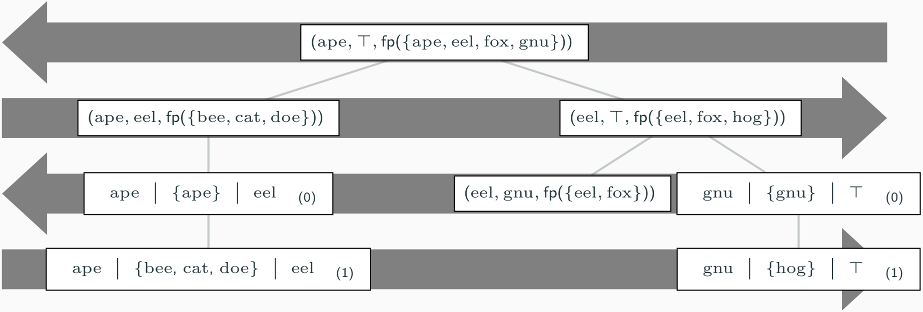
$(\mathrm{eel}, \mathrm{gnu}, \mathsf{fp}(\{\mathrm{eel}, \mathrm{fox}\}))$

$\mathrm{gnu} \mid \{\mathrm{gnu}\} \mid \top$ (0)

# Example Protocol Run

$$\text{ape} < \text{bee} < \text{cat} < \text{doe} < \text{eel} < \text{fox} < \text{gnu} < \text{hog} < \top$$

$$X_0 := \{\text{bee}, \text{cat}, \text{doe}, \text{eel}, \text{fox}, \text{hog}\} \qquad X_1 := \{\text{ape}, \text{eel}, \text{fox}, \text{gnu}\}$$



(ape, ⊤, fp({ape, eel, fox, gnu}))

(ape, eel, fp({bee, cat, doe}))

(eel, ⊤, fp({eel, fox, hog}))

ape | {ape} | eel (0)

(eel, gnu, fp({eel, fox}))

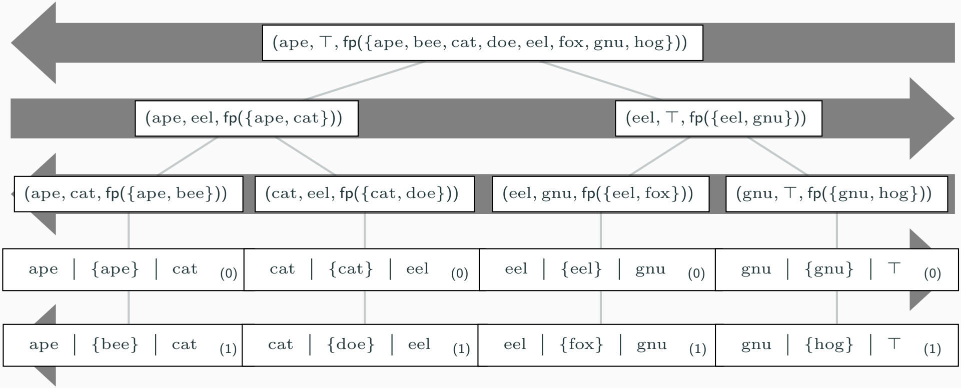
gnu | {gnu} | ⊤ (0)

ape | {bee, cat, doe} | eel (1)

gnu | {hog} | ⊤ (1)

# A Worst-Case Run

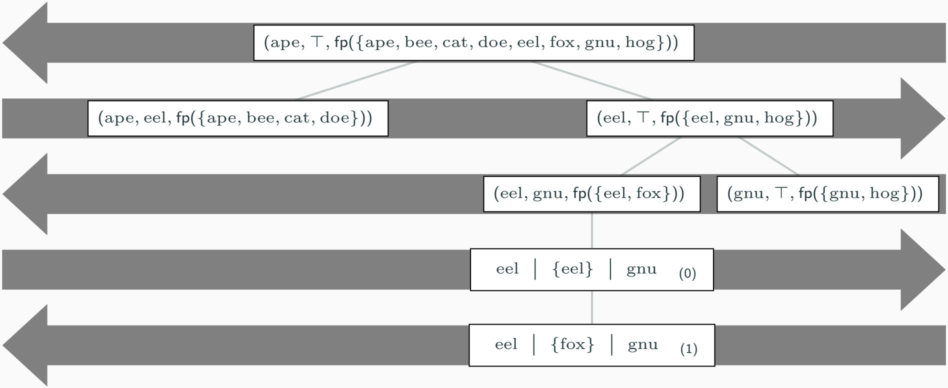
$$X_0 := \{\mathrm{ape}, \mathrm{cat}, \mathrm{eel}, \mathrm{gnu}\}$$
$$X_1 := \{\mathrm{ape}, \mathrm{bee}, \mathrm{cat}, \mathrm{doe}, \mathrm{eel}, \mathrm{fox}, \mathrm{gnu}, \mathrm{hog}\}$$



# A Different Worst-Case Run

$X_0 := \{\text{ape}, \text{bee}, \text{cat}, \text{doe}, \text{eel}, \text{gnu}, \text{hog}\}$

$X_1 := \{\text{ape}, \text{bee}, \text{cat}, \text{doe}, \text{eel}, \text{fox}, \text{gnu}, \text{hog}\}$



(ape, $\top$, fp({ape, bee, cat, doe, eel, fox, gnu, hog}))

(ape, eel, fp({ape, bee, cat, doe}))

(eel, $\top$, fp({eel, gnu, hog}))

(eel, gnu, fp({eel, fox}))

(gnu, $\top$, fp({gnu, hog}))

eel | {eel} | gnu (0)

eel | {fox} | gnu (1)