# 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

**Fingerprints**: Map sets into $\{0,1\}^k$ with low probability of collisions. Compare fingerprints for efficient probabilistic equality check.

# Basic Ideas

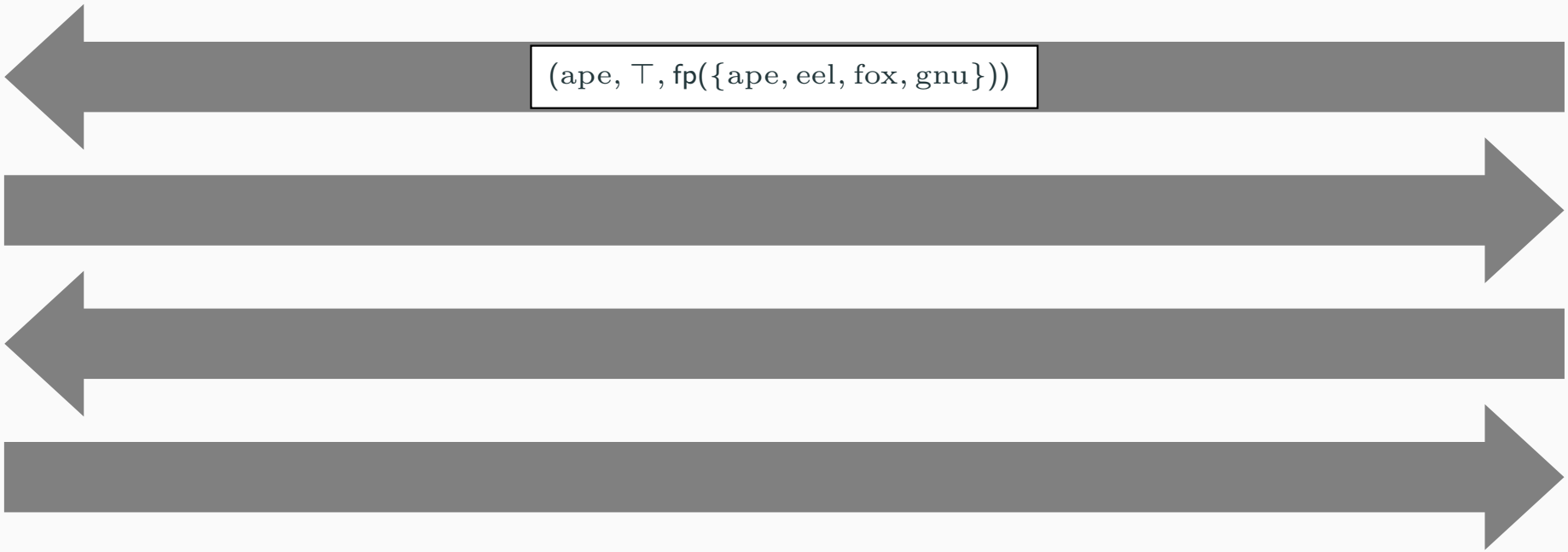
**Fingerprints**: Map sets into $\{0,1\}^k$ with low probability of collisions. Compare fingerprints for efficient probabilistic equality check.

**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 $X_0 \cup X_1 = (A_0 \cup B_0) \cup (A_1 \cup B_1)$.

# Example Protocol Run

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

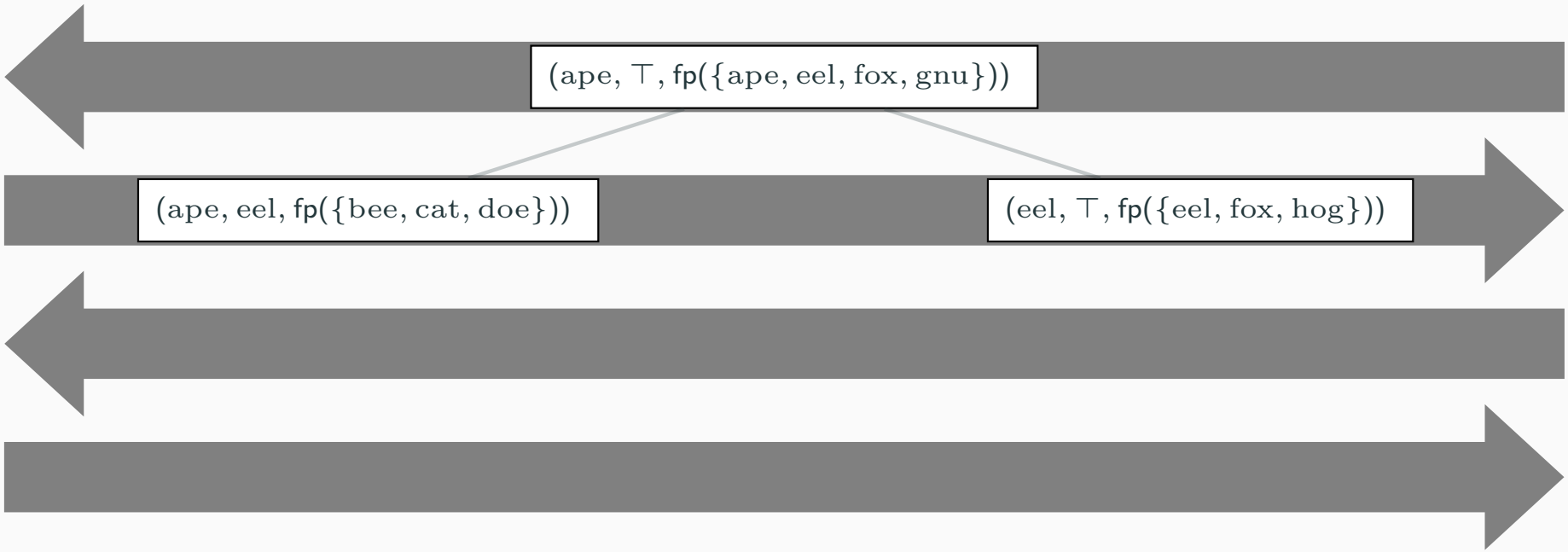
$X_0 := \{\text{bee}, \text{cat}, \text{doe}, \text{eel}, \text{fox}, \text{hog}\}$ $\quad$ $X_1 := \{\text{ape}, \text{eel}, \text{fox}, \text{gnu}\}$



# Example Protocol Run

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

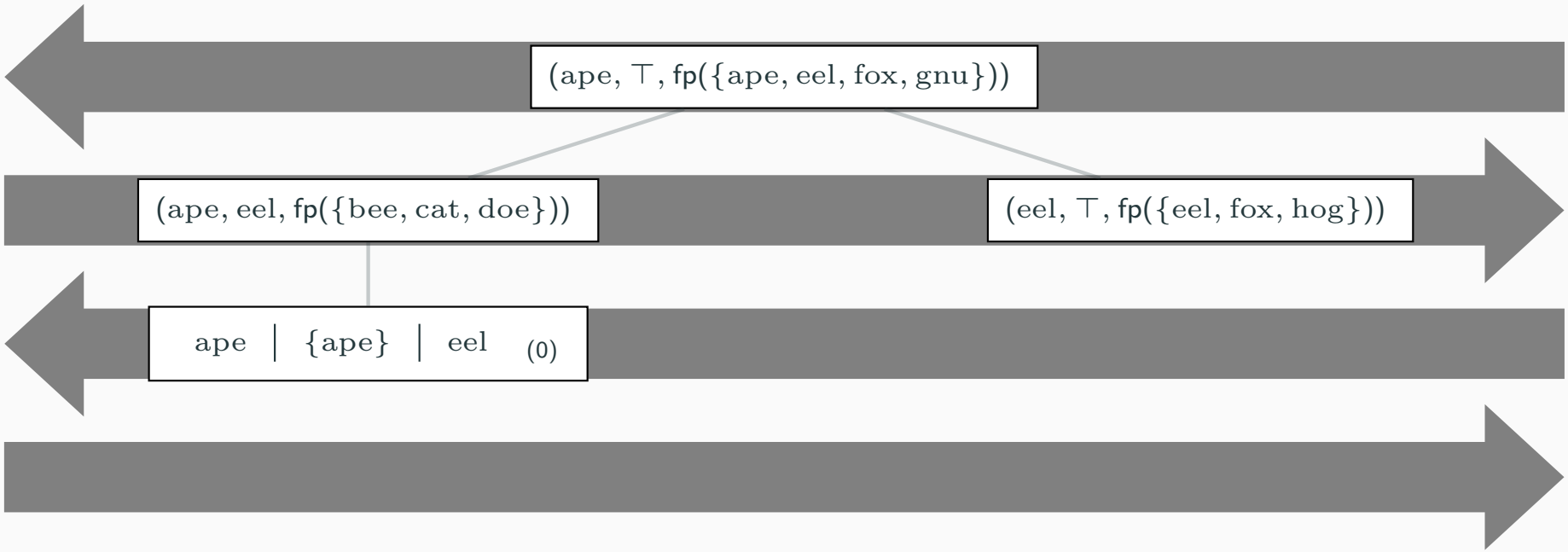
$X_0 := \{\mathrm{bee}, \mathrm{cat}, \mathrm{doe}, \mathrm{eel}, \mathrm{fox}, \mathrm{hog}\}$ $\quad$ $X_1 := \{\mathrm{ape}, \mathrm{eel}, \mathrm{fox}, \mathrm{gnu}\}$



# Example Protocol Run

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

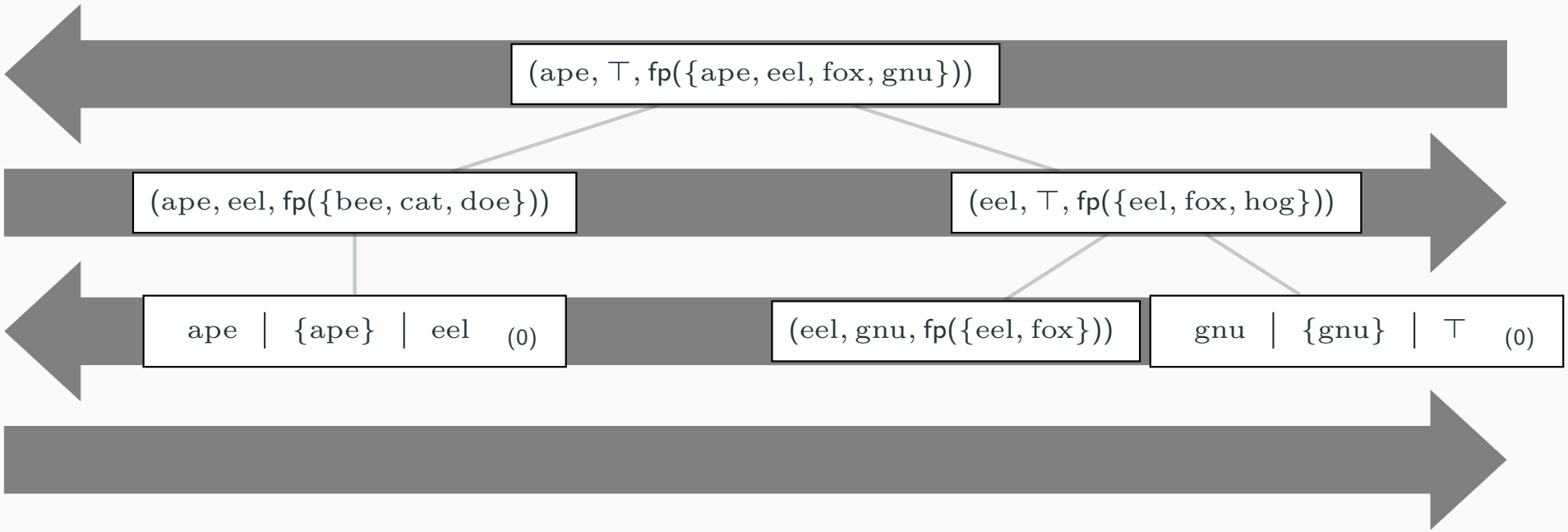
$$X_0 := \{\mathrm{bee}, \mathrm{cat}, \mathrm{doe}, \mathrm{eel}, \mathrm{fox}, \mathrm{hog}\} \qquad X_1 := \{\mathrm{ape}, \mathrm{eel}, \mathrm{fox}, \mathrm{gnu}\}$$



# Example Protocol Run

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

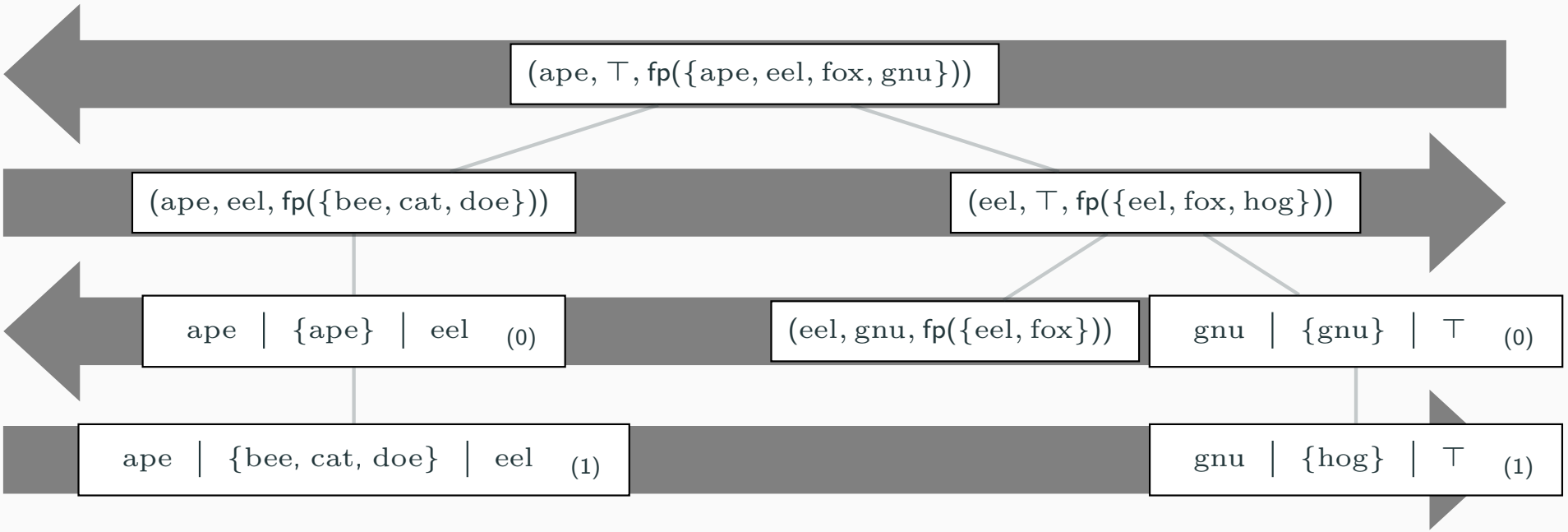
$X_0 := \{\text{bee}, \text{cat}, \text{doe}, \text{eel}, \text{fox}, \text{hog}\}$ $\quad$ $X_1 := \{\text{ape}, \text{eel}, \text{fox}, \text{gnu}\}$



# Example Protocol Run

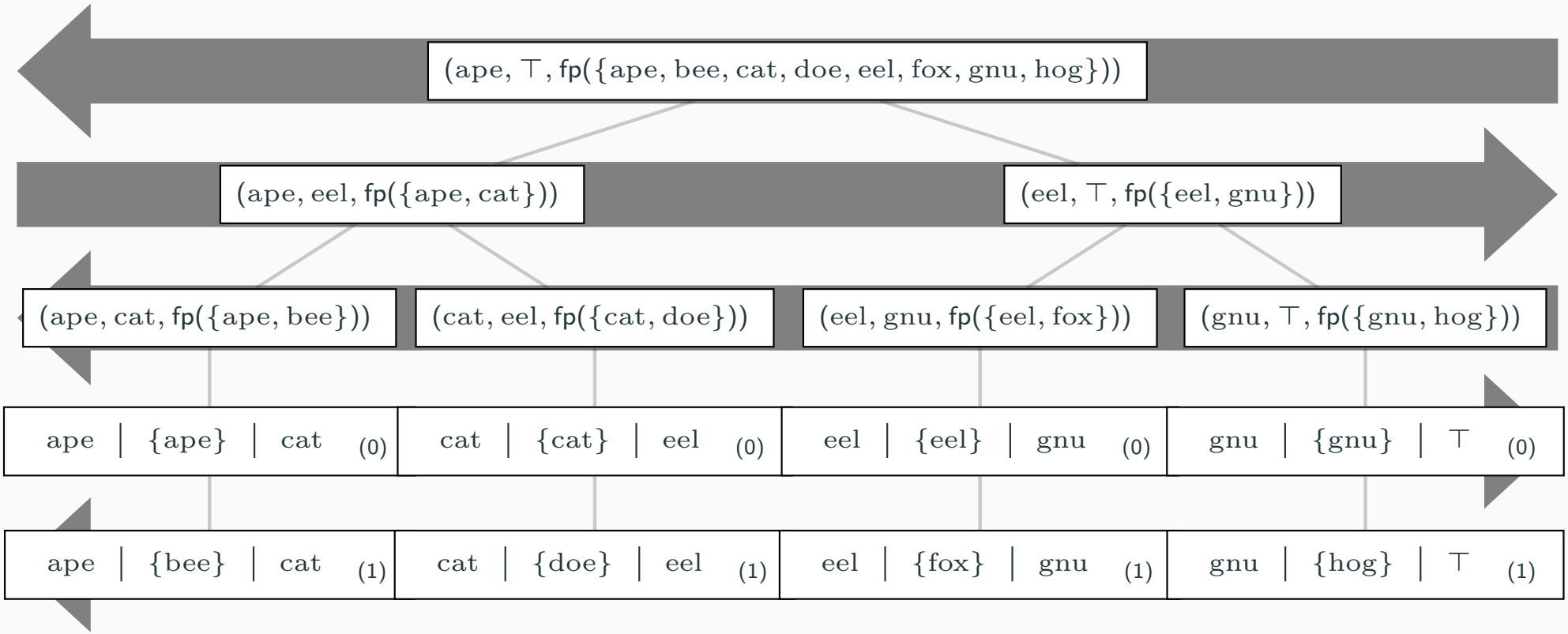
$$\text{ape} < \text{bee} < \text{cat} < \text{doe} < \text{eel} < \text{fox} < \text{gnu} < \text{hog} < \top$$

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



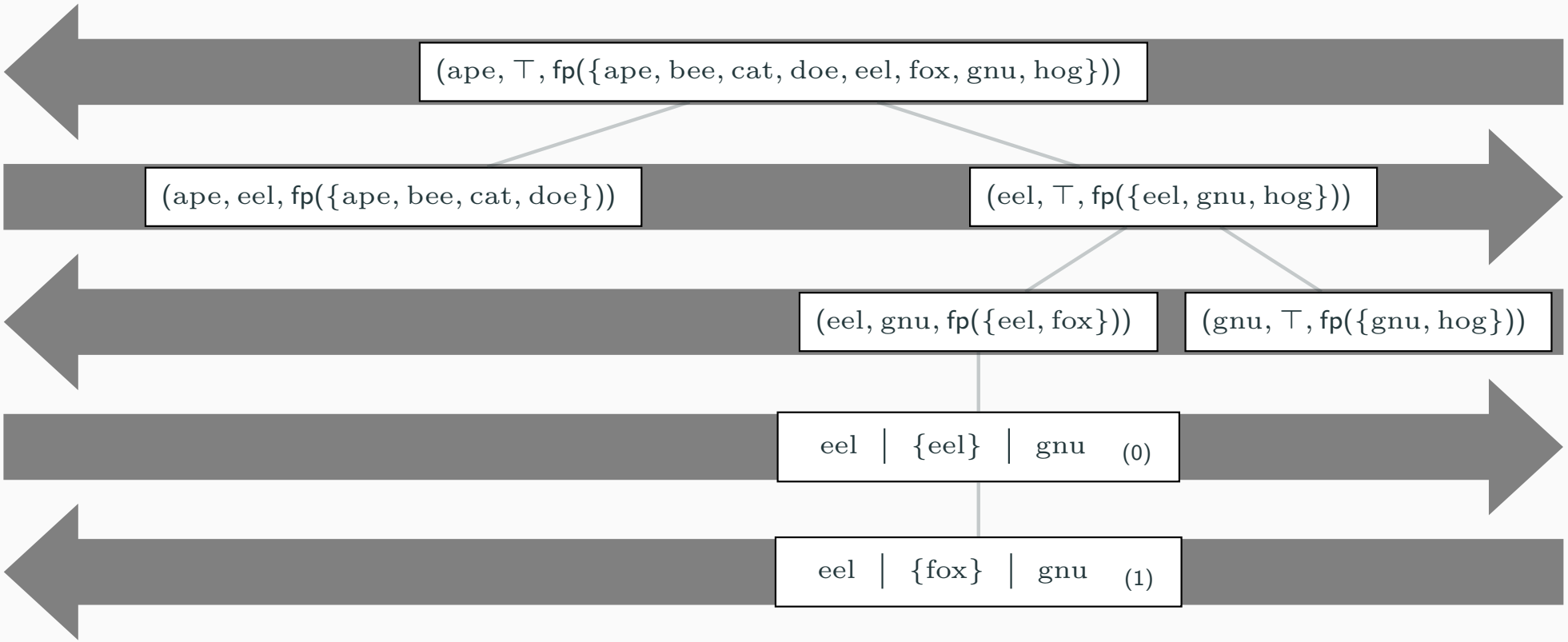
# A Worst-Case Run

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



# A Different Kind of Worst-Case Run

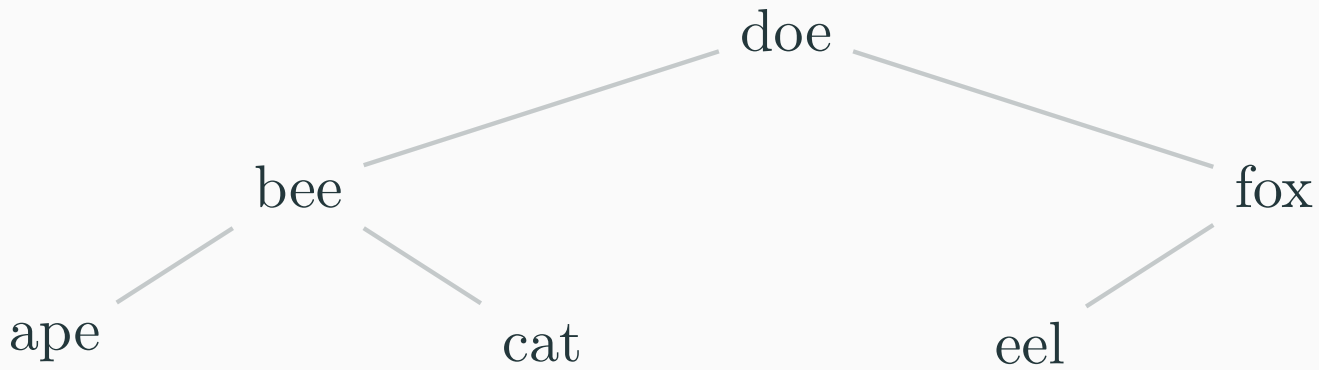
$$X_0 := \{\text{ape}, \text{bee}, \text{cat}, \text{doe}, \text{eel}, \text{gnu}, \text{hog}\}$$
$$X_1 := \{\text{ape}, \text{bee}, \text{cat}, \text{doe}, \text{eel}, \text{fox}, \text{gnu}, \text{hog}\}$$



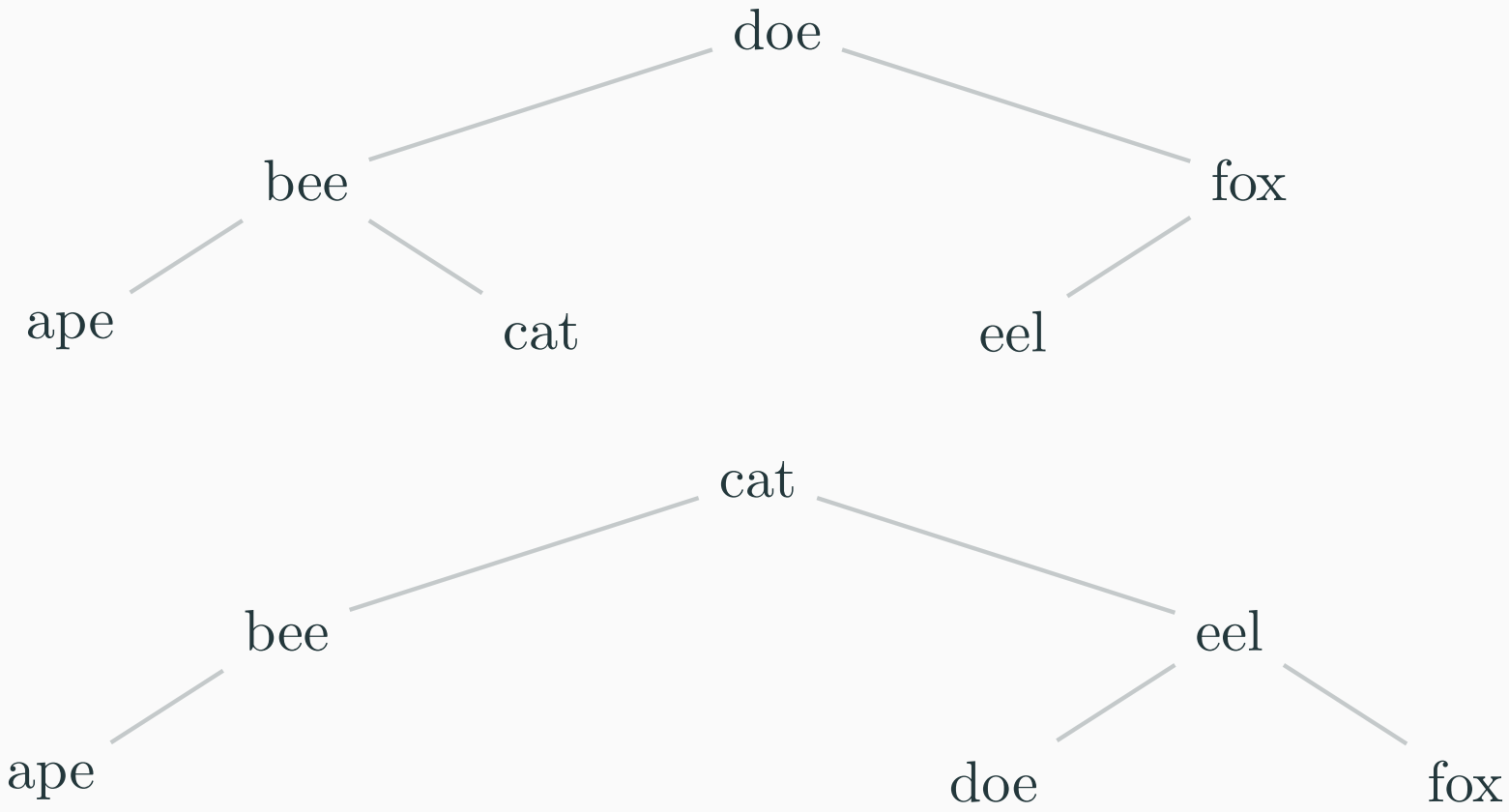
# Toward Suitable Fingerprints

$$((((\mathsf{h}(\mathrm{ape}) \oplus \mathsf{h}(\mathrm{bee})) \oplus \mathsf{h}(\mathrm{cat})) \oplus \mathsf{h}(\mathrm{doe})) \oplus \mathsf{h}(\mathrm{eel})) \oplus \mathsf{h}(\mathrm{fox})$$

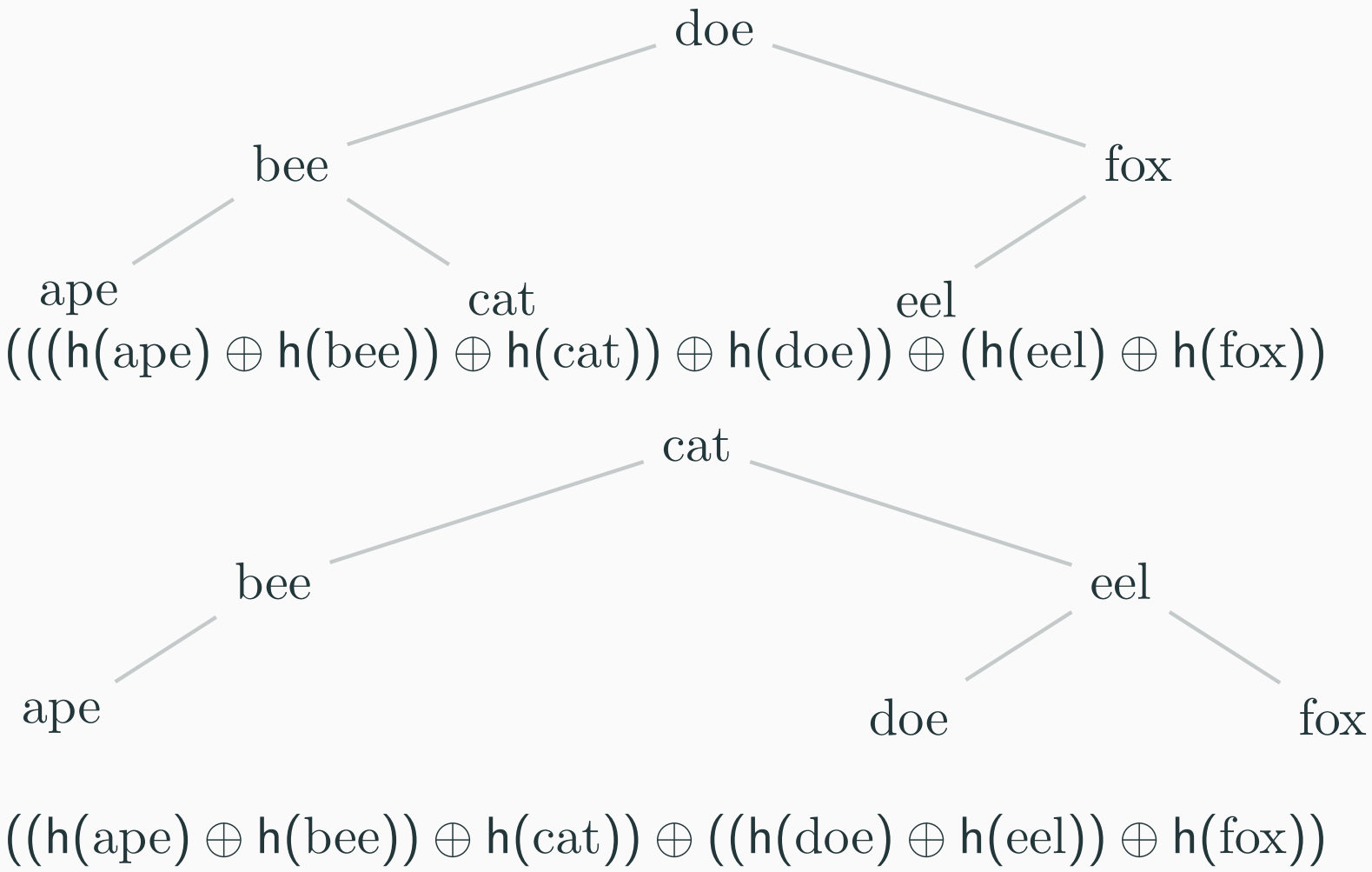
# Toward Suitable Fingerprints



# Toward Suitable Fingerprints



# Toward Suitable Fingerprints



# Monoidal Fingerprints

**Definition**
Let $M$ be a set, $\oplus : M \times M \to U$, and $\mathbb{0} \in M$.

We call $(U, \oplus, \mathbb{0})$ a *monoid* if it satisfies two properties:

**associativity:** for all $x, y, z \in M$: $(x \oplus y) \oplus z = x \oplus y \oplus z$

**neutral element:** for all $x \in M$: $\mathbb{0} \oplus x = x = x \oplus \mathbb{0}$.

# Monoidal Fingerprints

**Definition**
Let $M$ be a set, $\oplus : M \times M \to U$, and $\mathbb{0} \in M$.

We call $(U, \oplus, \mathbb{0})$ a *monoid* if it satisfies two properties:

**associativity:** for all $x, y, z \in M$: $(x \oplus y) \oplus z = x \oplus y \oplus z$

**neutral element:** for all $x \in M$: $\mathbb{0} \oplus x = x = x \oplus \mathbb{0}$.

**Definition**

Let $U$ be a set, $\preceq$ a linear order on $U$, $\mathcal{M} := (M, \oplus, \mathbb{0})$ a monoid, and $\mathrm{f} : U \to M$.

We *lift* f *to finite sets via* $\mathcal{M}$ to obtain $\mathrm{lift}_{\mathrm{f}}^{\mathcal{M}} : \mathcal{P}(U) \rightharpoonup M$ with:

$$\mathrm{lift}_{\mathrm{f}}^{\mathcal{M}}(\emptyset) := \mathbb{0}$$
$$\mathrm{lift}_{\mathrm{f}}^{\mathcal{M}}(S) := \mathrm{f}(\{\min(S)\}) \oplus \mathrm{lift}_{\mathrm{f}}^{\mathcal{U}}(S \setminus \{\min(S)\})$$

# Example Tree