Eris

Resourceful error bound reasoning for higher-order probabilistic programs

Alejandro Aguirre
Philipp Haselwarter
Markus de Medeiros
Kwing Hei Li
Simon Oddershede Gregersen
Joseph Tassarotti
Lars Birkedal

AARHUS UNIVERSITY
NYU

Approximate Specifications

$$\text{hash} \; : A \to \text{int64}$$

$$\text{collide} \; : A \to A \to \text{bool}$$

$$\text{collide } x \; y = (\text{hash } x = \text{hash } y)$$

Approximate Specifications

$$\mathsf{hash} \ : A \to \mathsf{int64}$$

$$\mathsf{collide} \ : A \to A \to \mathsf{bool}$$

$$\mathsf{collide}\ x\ y = (\mathsf{hash}\ x = \mathsf{hash}\ y)$$

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{\approx}$$

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Useful reasoning principles,

Union Bound

$$\frac{\{P\}\, e_1\, \{Q\}_{\epsilon_1} \qquad \{Q\}\, e_2\, \{R\}_{\epsilon_2}}{\{P\}\, e_1; e_2\, \{R\}_{\epsilon_1 + \epsilon_2}}$$

Sampling

$$\frac{\Pr_{x \sim D}[x \notin S] < \epsilon}{\{\mathsf{True}\}\ \mathsf{sample}(D)\ \{x.\, x \in S\}_{\epsilon}}$$

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Useful reasoning principles,

Union Bound

$$\frac{\{P\}\, e_1\, \{Q\}_{\epsilon_1} \qquad \{Q\}\, e_2\, \{R\}_{\epsilon_2}}{\{P\}\, e_1; e_2\, \{R\}_{\epsilon_1 + \epsilon_2}}$$

Sampling

$$\frac{\Pr_{x \sim D}[x \notin S] < \epsilon}{\{\mathsf{True}\}\ \mathsf{sample}(D)\ \{x.\, x \in S\}_{\epsilon}}$$

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Useful reasoning principles,

Union Bound

$$\frac{\{P\}\, e_1\, \{Q\}_{\epsilon_1} \qquad \{Q\}\, e_2\, \{R\}_{\epsilon_2}}{\{P\}\, e_1; e_2\, \{R\}_{\epsilon_1 + \epsilon_2}}$$

Sampling

$$\frac{\Pr_{x \sim D}[x \notin S] < \epsilon}{\{\mathsf{True}\}\ \mathsf{sample}(D)\ \{x.\, x \in S\}_{\epsilon}}$$

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Useful reasoning principles, but limited compositionality.

Approximate Specifications

aHL

$$\{x \neq y\}\ \textsf{collide}\ x\ y\ \{b.\, b = \textsf{false}\}_{2^{-64}}$$

Useful reasoning principles, **but limited compositionality.**

Limitation 1

$$\frac{\forall a.\, \{\ldots\}\ f\ a\ \{\ldots\}_{\epsilon(a)}}{\{\ldots\}\ \textsf{map}\ f\ L\ \{\ldots\}_{?}}$$

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Useful reasoning principles, **but limited compositionality.**

Limitation 1

$$\frac{\forall a.\, \{\ldots\}\ f\ a\ \{\ldots\}_{\epsilon(a)}}{\{\ldots\}\ \mathsf{map}\ f\ L\ \{\ldots\}_{\sum_{a \in L} \epsilon(a)}}$$

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Useful reasoning principles, **but limited compositionality.**

Limitation 1

$$\frac{\forall a.\, \{\ldots\}\ f\ a\ \{\ldots\}_{\epsilon(a)}}{\{\ldots\}\ \mathsf{map}\ f\ L\ \{\ldots\}_{\sum_{a \in L} \epsilon(a)}}$$

error specifications propagate

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Useful reasoning principles, **but limited compositionality.**

Limitation 2

$$\{\top\}\ G\ d\ \{d.\, P\}_{0}$$
$$\{\top\}\ F\ d\ \{d.\, P\}_{1/100}$$

$$\begin{aligned}\mathsf{test}\ d = \ & \mathsf{if}\ \mathsf{decide}\ d \\ & \mathsf{then}\ (\mathsf{true}, G\ d) \\ & \mathsf{else}\ (\mathsf{false}, F\ d)\end{aligned}$$

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Useful reasoning principles, **but limited compositionality.**

Limitation 2

$$\{\top\}\ G\ d\ \{d.\, P\}_{0}$$
$$\{\top\}\ F\ d\ \{d.\, P\}_{1/100}$$

$$\mathsf{test}\ d = \ \mathsf{if\ decide}\ d\ \mathsf{then}\ (\mathsf{true}, G\ d)\ \mathsf{else}\ (\mathsf{false}, F\ d)$$

Approximate Specifications

aHL

$$\{x \neq y\}\ \text{collide}\ x\ y\ \{b.\, b = \text{false}\}_{2^{-64}}$$

Useful reasoning principles, **but limited compositionality.**

Limitation 2

$$\{\top\}\ G\ d\ \{d.\, P\}_0$$

$$\{\top\}\ F\ d\ \{d.\, P\}_{1/100}$$

$$\begin{aligned}\text{test}\ d = \ & \text{if decide}\ d \\ & \text{then}\ (\text{true}, G\ \ d) \\ & \text{else}\ (\text{false}, F\ \ d)\end{aligned}$$

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Useful reasoning principles, **but limited compositionality.**

Limitation 2

$$\{\top\}\ G\ d\ \{d.\, P\}_{0}$$
$$\{\top\}\ F\ d\ \{d.\, P\}_{1/100}$$

$$\mathsf{test}\ d = \ \mathsf{if}\ \mathsf{decide}\ d\ \mathsf{then}\ (\mathsf{true}, G\ d)\ \mathsf{else}\ (\mathsf{false}, F\ d)$$

$$\{\top\}\ \mathsf{test}\ d\ \{(v, d).\, P\}_{?}$$

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Useful reasoning principles, **but limited compositionality.**

Limitation 2

$$\{\top\}\ G\ d\ \{d.\, P\}_0$$

$$\{\top\}\ F\ d\ \{d.\, P\}_{1/100}$$

$$\mathsf{test}\ d = \ \mathsf{if\ decide}\ d\ \ \mathsf{then}\ (\mathsf{true}, G\ d)\ \ \mathsf{else}\ (\mathsf{false}, F\ d)$$

$$\{\top\}\ \mathsf{test}\ d\ \{(v, d).\, P\}_?$$

error depends on return value

Approximate Specifications

aHL

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

Error Credits

Eris

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

$$\{↯(2^{-64}) * x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}$$

Altes Museum, Public domain, via Wikimedia Commons

Error Credits

Eris

$$\{x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}_{2^{-64}}$$

$$\{\lightning(2^{-64}) * x \neq y\}\ \mathsf{collide}\ x\ y\ \{b.\, b = \mathsf{false}\}$$

Altes Museum, Public domain, via Wikimedia Commons

Expected Error Bounds as a Resource

Error Credits

Eris

Expected Error Bounds as a Resource

$$\{ ↯(\epsilon) * P \} \; f \; \{ Q \}$$

Iris

Error Credits

Eris

Expected Error Bounds as a Resource

$$\boxed{\{\lightning(\epsilon) * P\}\ f\ \{Q\}}$$

$$\frac{\{P\}\ f\ \{Q\}}{\{P * \lightning(\epsilon)\}\ f\ \{Q * \lightning(\epsilon)\}}$$

$$\boxed{\lightning(\epsilon)} \vdash P$$

$$\Big\{\ \{P * \lightning(\epsilon)\}\ f\ \{Q\}\ \Big\}\ g\ \{R\}$$

Iris

The Eris Logic

Limitation 1

The Eris Logic

Limitation 1

Standard higher-order specification:

$$\frac{\forall a,\ \{P\ a\}\ f\ a\ \{Q\ a\}}{\left\{\bigstar_{a \in L} (P\ a)\right\} \mathsf{map}\ f\ L \left\{L'.\ \bigstar_{a \in L'} (Q\ a)\right\}}$$

The Eris Logic

Limitation 1

Standard higher-order specification:

$$\frac{\forall a,\ \{P\ a\}\ f\ a\ \{Q\ a\}}{\left\{ \mathop{\Large\ast}_{a \in L} (P\ a) \right\} \ \mathsf{map}\ f\ L \left\{ L'.\ \mathop{\Large\ast}_{a \in L'} (Q\ a) \right\}}$$

Derived error-aware specification:

$$\frac{\forall y,\ \{ ↯ (2^{-64}) \}\ \mathsf{hash}\ y\ \{v.\, v \neq v'\}}{\left\{ \mathop{\Large\ast}_{a \in L} ↯ (2^{-64}) \right\} \ \mathsf{map\ hash}\ L \left\{ L'.\ \mathop{\Large\ast}_{a \in L'} a \neq v' \right\}}$$

The Eris Logic

Limitation 2

$$\{\top\}\, G\ d\, \{d.\, P\}_{0}$$
$$\{\top\}\, F\ d\, \{d.\, P\}_{1/100}$$

$$\begin{aligned}\text{test } d = \ & \text{if decide } d \\ & \text{then } (\text{true}, G\ d) \\ & \text{else } (\text{false}, F\ d)\end{aligned}$$

$$\{\top\}\, \text{test } d\, \{(v, d).\, P\}_{?}$$

The Eris Logic

Limitation 2

$$\{\top\}\ G\ d\ \{d.\ P\}$$
$$\{⚡(1/100)\}\ F\ d\ \{d.\ P\}$$

$$\text{test } d = \ \text{if decide } d \ \text{then } (\text{true}, G\ d) \ \text{else } (\text{false}, F\ d)$$

State-dependent specification:

$$\left\{ ⚡(1/100) \right\} \text{test } d \left\{ (v, d).\ P * \left(\begin{array}{l} \text{if } v \\ \text{then } ⚡(1/100) \\ \text{else } \top \end{array} \right) \right\}$$

Error Credits

Core Rules

Error Credits

Core Rules

Spending $\lightning(1) \vdash \bot$

Error Credits

Core Rules

Spending $\lightning(1) \vdash \bot$

Splitting $\lightning(\epsilon_1 + \epsilon_2) \dashv\vdash \lightning(\epsilon_1) * \lightning(\epsilon_2)$

Error Credits

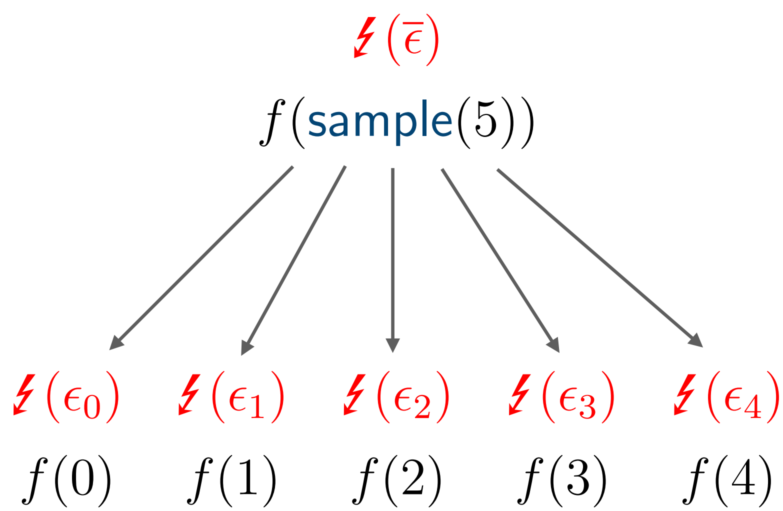
Core Rules

Spending $\lightning(1) \vdash \bot$

Splitting $\lightning(\epsilon_1 + \epsilon_2) \dashv\vdash \lightning(\epsilon_1) * \lightning(\epsilon_2)$

Averaging

$$\frac{\mathbb{E}_{x \sim D}[\epsilon_x] = \overline{\epsilon}}{\{\lightning(\overline{\epsilon})\}\ \mathsf{sample}(D)\ \{x.\, \lightning(\epsilon_x)\}}$$



Error Credits

Derived Rules

aHL Union Bound

$$\frac{\{P\}\, e_1\, \{Q\}_{\epsilon_1} \qquad \{Q\}\, e_2\, \{R\}_{\epsilon_2}}{\{P\}\, e_1; e_2\, \{R\}_{\epsilon_1 + \epsilon_2}}$$

Error Credits

Derived Rules

$$\{\lightning(\epsilon_1) * P\}\, e_1\, \{Q\}$$

$$\{\lightning(\epsilon_2) * Q\}\, e_2\, \{R\}$$

aHL Union Bound

$$\frac{\{P\}\, e_1\, \{Q\}_{\epsilon_1} \qquad \{Q\}\, e_2\, \{R\}_{\epsilon_2}}{\{P\}\, e_1; e_2\, \{R\}_{\epsilon_1 + \epsilon_2}}$$

Error Credits

Derived Rules

aHL Union Bound

$$\frac{\{P\}\, e_1\, \{Q\}_{\epsilon_1} \qquad \{Q\}\, e_2\, \{R\}_{\epsilon_2}}{\{P\}\, e_1; e_2\, \{R\}_{\epsilon_1 + \epsilon_2}}$$

$$\{ \lightning(\epsilon_1) * P\}\, e_1\, \{Q\}$$

$$\{ \lightning(\epsilon_2) * Q\}\, e_2\, \{R\}$$

$$\lightning(\epsilon_1 + \epsilon_2) * P$$

$$e_1;\ e_2$$

Error Credits

Derived Rules

aHL Union Bound

$$\frac{\{P\}\, e_1\, \{Q\}_{\epsilon_1} \qquad \{Q\}\, e_2\, \{R\}_{\epsilon_2}}{\{P\}\, e_1; e_2\, \{R\}_{\epsilon_1 + \epsilon_2}}$$

$$\{\lightning(\epsilon_1) * P\}\, e_1\, \{Q\}$$

$$\{\lightning(\epsilon_2) * Q\}\, e_2\, \{R\}$$

$$\lightning(\epsilon_1) * \lightning(\epsilon_2) * P$$

$$e_1;\ e_2$$

Splitting

Error Credits

Derived Rules

aHL Union Bound

$$\frac{\{P\}\, e_1\, \{Q\}_{\epsilon_1} \qquad \{Q\}\, e_2\, \{R\}_{\epsilon_2}}{\{P\}\, e_1; e_2\, \{R\}_{\epsilon_1 + \epsilon_2}}$$

$$\lightning(\epsilon_1) * \lightning(\epsilon_2) * P$$

Splitting

$$\{\lightning(\epsilon_2) * Q\}\, e_2\, \{R\}$$

$$e_1;\ e_2$$

$$\{\lightning(\epsilon_1) * P\}\, e_1\, \{Q\}$$

Frame Rule

$$\lightning(\epsilon_2) * Q$$

$$e_2$$

Error Credits

Derived Rules

aHL Union Bound

$$\frac{\{P\}\, e_1\, \{Q\}_{\epsilon_1} \qquad \{Q\}\, e_2\, \{R\}_{\epsilon_2}}{\{P\}\, e_1; e_2\, \{R\}_{\epsilon_1 + \epsilon_2}}$$

$$↯(\epsilon_1) * ↯(\epsilon_2) * P$$

Splitting

$$e_1;\ e_2$$

Frame Rule

$$\{↯(\epsilon_1) * P\}\, e_1\, \{Q\}$$

$$↯(\epsilon_2) * Q$$

$$e_2$$

$$\{↯(\epsilon_2) * Q\}\, e_2\, \{R\}$$

$$R$$

Error Credits

Derived Rules

aHL Sampling

$$\frac{\Pr_{x \sim D}[x \notin S] < \epsilon}{\{\mathsf{True}\}\ \mathsf{sample}(D)\ \{x.\, x \in S\}_{\epsilon}}$$

Error Credits

Derived Rules

aHL Sampling

$$\frac{\Pr_{x \sim D}[x \notin S] < \epsilon}{\{\mathsf{True}\}\ \mathsf{sample}(D)\ \{x.\, x \in S\}_{\epsilon}}$$

$$↯(1/5)$$

$$f(\mathsf{sample}(5))$$

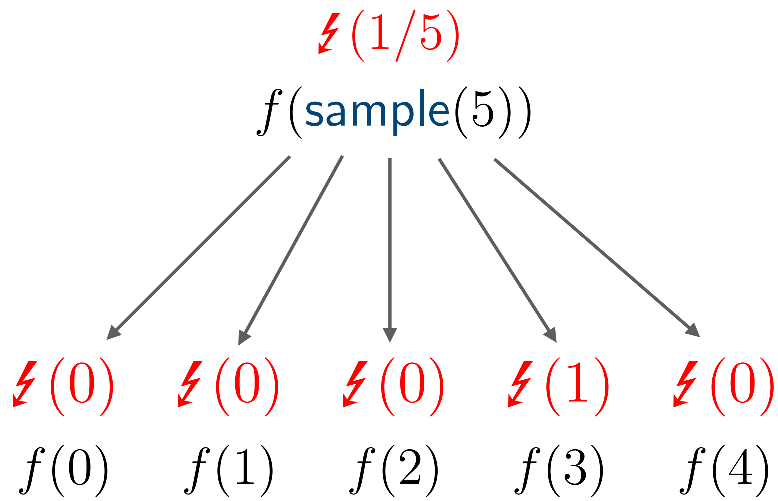
Error Credits

Derived Rules

aHL Sampling

$$\frac{\Pr_{x \sim D}[x \notin S] < \epsilon}{\{\mathsf{True}\}\ \mathsf{sample}(D)\ \{x.\, x \in S\}_{\epsilon}}$$

Averaging

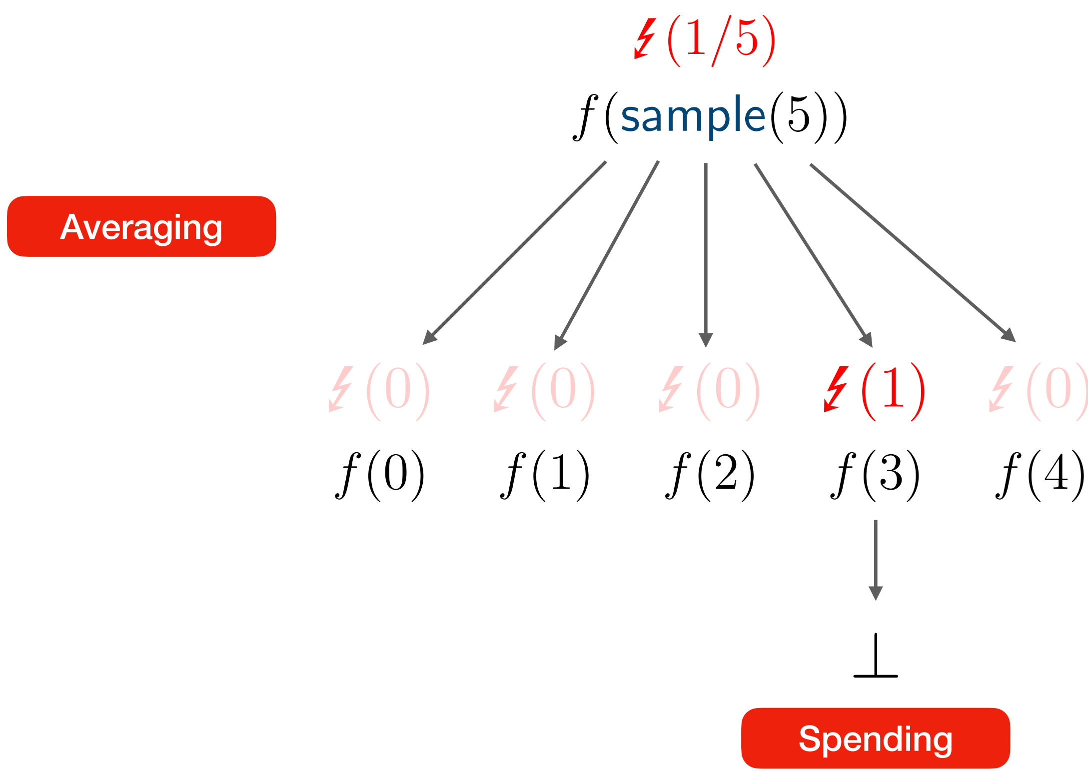


Error Credits

Derived Rules

aHL Sampling

$$\frac{\Pr_{x \sim D}[x \notin S] < \epsilon}{\{\mathsf{True}\}\ \mathsf{sample}(D)\ \{x.\, x \in S\}_{\epsilon}}$$

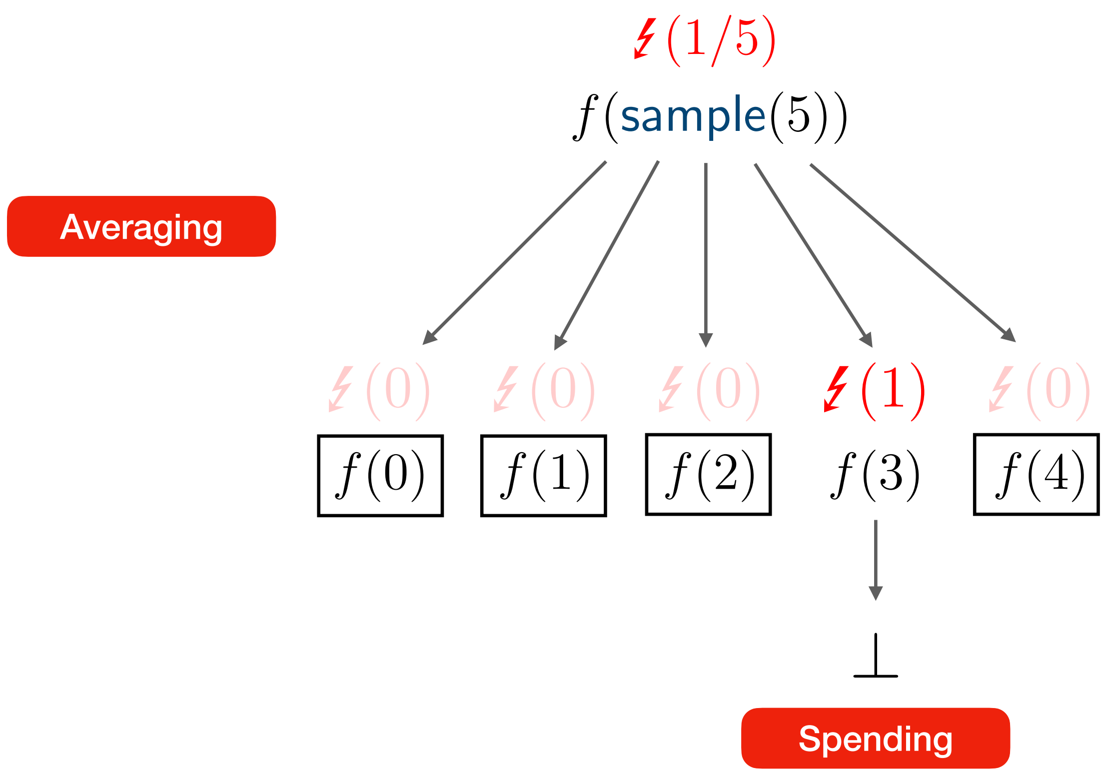


Error Credits

Derived Rules

aHL Sampling

$$\frac{\Pr_{x \sim D}[x \notin S] < \epsilon}{\{\mathsf{True}\}\ \mathsf{sample}(D)\ \{x.\, x \in S\}_{\epsilon}}$$



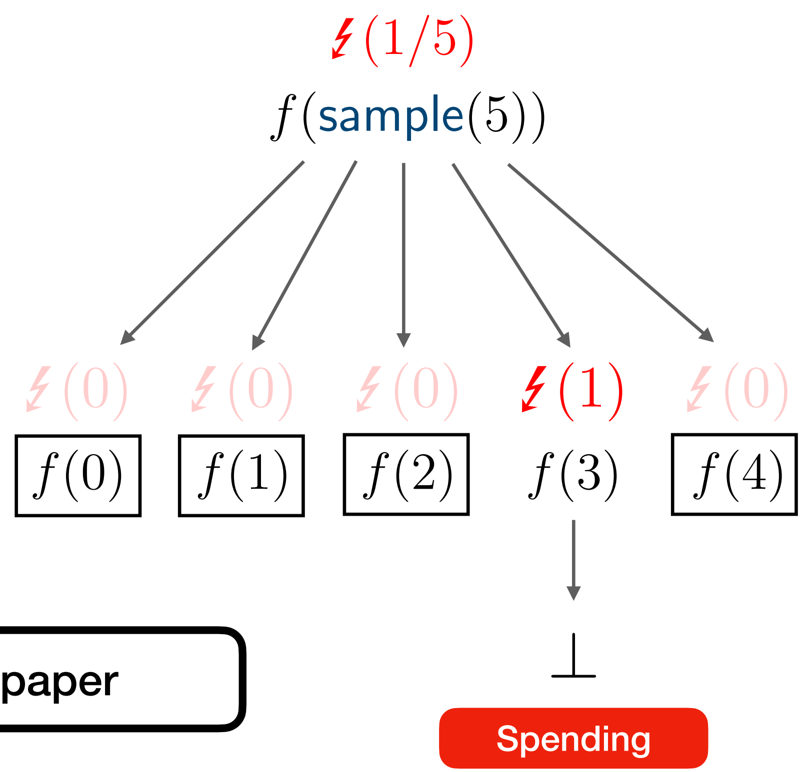
Error Credits

Derived Rules

aHL Sampling

$$\frac{\Pr_{x \sim D}[x \notin S] < \epsilon}{\{\mathsf{True}\}\ \mathsf{sample}(D)\ \{x.\, x \in S\}_{\epsilon}}$$

Averaging



More derived rules in paper

Hash-based authentication in Eris

Hash Collisions

$$\text{hash} : A \rightarrow \text{int64}$$

$$\begin{aligned}\text{hash } x = \ & \text{match get } x \text{ with} \\ & \quad \text{Some } (v) \Rightarrow v \\ & \mid \text{None} \Rightarrow \ \text{let } v = \text{sample}(2^{64}) \text{ in} \\ & \qquad\qquad\quad \text{set } x\ v; \\ & \qquad\qquad\quad v \\ & \text{end}\end{aligned}$$

Hash Collisions

$$\text{hash} : A \to \text{int64}$$

$$\begin{aligned}\text{hash } x = \;& \text{match get } x \text{ with} \\ & \quad \text{Some } (v) \Rightarrow v \\ & \mid \text{None} \Rightarrow \; \text{let } v = \text{sample}(2^{64}) \text{ in} \\ & \qquad\qquad \text{set } x \; v; \\ & \qquad\qquad v \\ & \text{end}\end{aligned}$$

Property: collisionFree N

- **Map is collision-free**
- **At most N hashes**

Hash Collisions

hash : $A \to$ int64

$$
\begin{aligned}
\text{hash } x = \;& \text{match get } x \text{ with} \\
& \quad \text{Some } (v) \Rightarrow v \\
& \mid \text{None} \Rightarrow \;\; \text{let } v = \text{sample}(2^{64}) \text{ in} \\
& \qquad\qquad\quad\; \text{set } x \ v; \\
& \qquad\qquad\quad\; v \\
& \text{end}
\end{aligned}
$$

Property: collisionFree N

Hash Collisions

$$\left\{ \begin{array}{c} \text{collisionFree } N \ * \\ \lightning(?) \end{array} \right\} \text{hash } x \left\{ v. \begin{array}{c} \text{collisionFree } (N+1) \ * \\ \text{get } x = v \end{array} \right\}$$

hash : $A \to$ int64

hash $x =$ match get x with
  Some $(v) \Rightarrow v$
  | None $\Rightarrow$ let $v =$ sample(2^{64}) in
    set x v;
    v
end

Property: collisionFree N

Hash Collisions

$$\text{hash} : A \rightarrow \text{int64}$$

```
hash x =  match get x with
            Some (v) ⇒ v
          | None ⇒  let v = sample(2^64) in
                    set x v;
                    v
          end
```

$$\left\{ \begin{matrix} \text{collisionFree } N \ast \\ ⚡(?) \end{matrix} \right\} \text{hash } x \left\{ v. \begin{matrix} \text{collisionFree } (N+1) \ast \\ \text{get } x = v \end{matrix} \right\}$$

- **Already Hashed**

Property: collisionFree N

Hash Collisions

hash : $A \to$ int64

hash $x =$ match get x with
  Some $(v) \Rightarrow v$
  | None $\Rightarrow$ let $v =$ sample(2^{64}) in
    set x v;
    v
  end

$$\left\{ \begin{array}{c} \text{collisionFree } N \ * \\ ⚡(?) \end{array} \right\} \text{hash } x \left\{ v. \begin{array}{c} \text{collisionFree } (N+1) \ * \\ \text{get } x = v \end{array} \right\}$$

- **Already Hashed**

  ⚡(0)

Property: collisionFree N

Hash Collisions

hash : $A \to$ int64

hash $x =$ match get x with
  Some $(v) \Rightarrow v$
  | None $\Rightarrow$ let $v =$ sample(2^{64}) in
    set x v;
    v
end

$$\left\{ \begin{array}{c} \text{collisionFree } N \,* \\ ⚡(?) \end{array} \right\} \text{hash } x \left\{ v. \begin{array}{c} \text{collisionFree } (N+1) \,* \\ \text{get } x = v \end{array} \right\}$$

- **Already Hashed**

  ⚡(0)

- **New Hash**

Property: collisionFree N

Hash Collisions

$\text{hash} : A \to \text{int64}$

$$\begin{aligned}\text{hash}\ x = \ & \text{match get}\ x\ \text{with} \\ & \quad \text{Some}\ (v) \Rightarrow v \\ & \mid \text{None} \Rightarrow \boxed{\text{let}\ v = \text{sample}(2^{64})\ \text{in}} \\ & \qquad\qquad \text{set}\ x\ v; \\ & \qquad\qquad v \\ & \text{end}\end{aligned}$$

$$\left\{ \begin{matrix} \text{collisionFree}\ N * \\ \lightning(?) \end{matrix} \right\} \text{hash}\ x \left\{ v.\ \begin{matrix} \text{collisionFree}\ (N+1) * \\ \text{get}\ x = v \end{matrix} \right\}$$

- **Already Hashed**

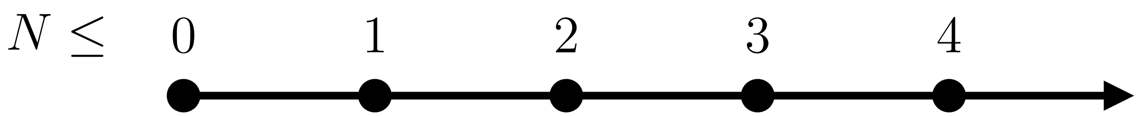
  $\lightning(0)$

- **New Hash**

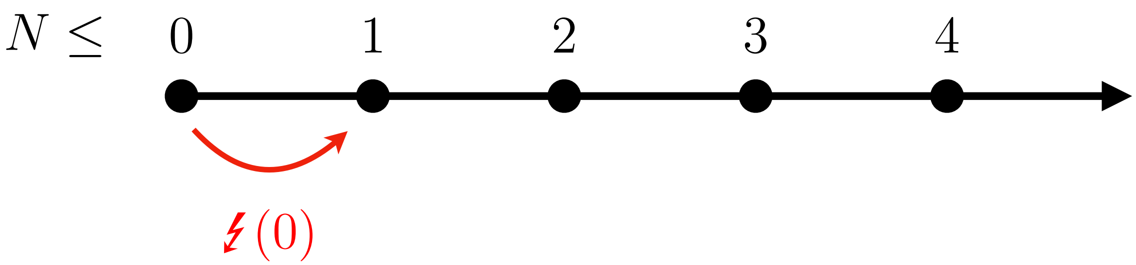
  $\lightning(?)$

Property: collisionFree N

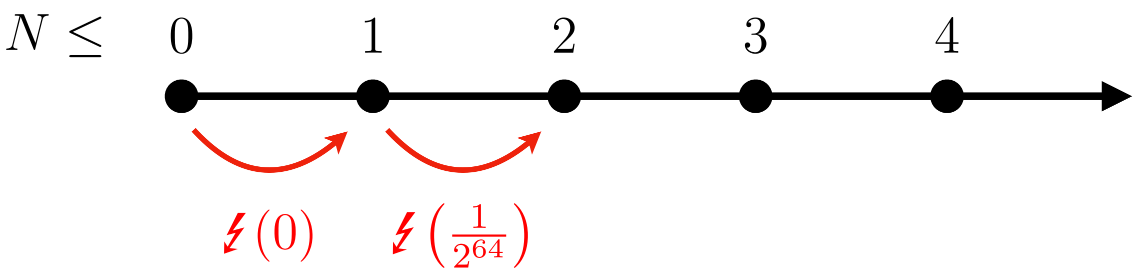
Credit Arithmetic



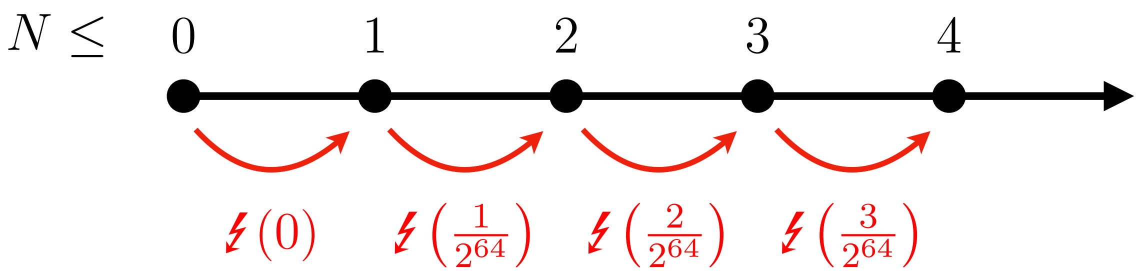
Credit Arithmetic



Credit Arithmetic



Credit Arithmetic



Hash Collisions

$$\left\{ \begin{array}{c} \text{collisionFree } N * \\ ⚡(?) \end{array} \right\} \text{hash } x \left\{ v. \begin{array}{c} \text{collisionFree } (N+1) * \\ \text{get } x = v \end{array} \right\}$$

```
hash : A → int64
hash x =  match get x with
            Some (v) ⇒ v
          | None ⇒ let v = sample(2^64) in
                   set x v;
                   v
          end
```

- **Already Hashed**

  $⚡(0)$

- **New Hash**

  $⚡\left(\frac{N}{2^{64}}\right)$

Property: collisionFree N

Hash Collisions

$\text{hash} : A \to \text{int64}$

```
hash x =  match get x with
            Some (v) ⇒ v
          | None ⇒  let v = sample(2^64) in
                    set x v;
                    v
          end
```

$$\left\{ \begin{array}{c} \text{collisionFree } N \, * \\ ↯(N \cdot 2^{-64}) \end{array} \right\} \text{hash } x \left\{ v. \begin{array}{c} \text{collisionFree } (N+1) \, * \\ \text{get } x = v \end{array} \right\}$$

Property: collisionFree N

Hash Collisions

hash : $A \rightarrow$ int64

hash $x =$ match get x with
  Some $(v) \Rightarrow v$
  | None $\Rightarrow$ let $v =$ sample(2^{64}) in
    set x v;
    v
  end

$$\left\{ \begin{array}{c} \text{collisionFree } N \; * \\ ↯ (N \cdot 2^{-64}) \end{array} \right\} \text{hash } x \left\{ v. \begin{array}{c} \text{collisionFree } (N+1) \; * \\ \text{get } x = v \end{array} \right\}$$

Simplify client dependency on N?

Property: collisionFree N

Hash Collisions

hash : $A \to$ int64

```
hash x =  match get x with
            Some (v) ⇒ v
          | None ⇒  let v = sample(2^64) in
                    set x v;
                    v
          end
```

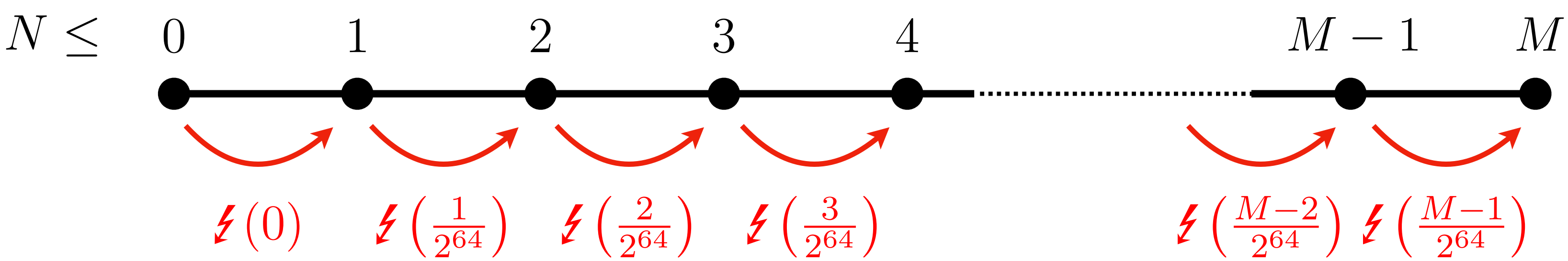
$$\left\{ \begin{array}{c} \text{collisionFree } N \,* \\ ↯(N \cdot 2^{-64}) \end{array} \right\} \text{hash } x \left\{ v. \begin{array}{c} \text{collisionFree } (N+1) \,* \\ \text{get } x = v \end{array} \right\}$$

Simplify client dependency on N?

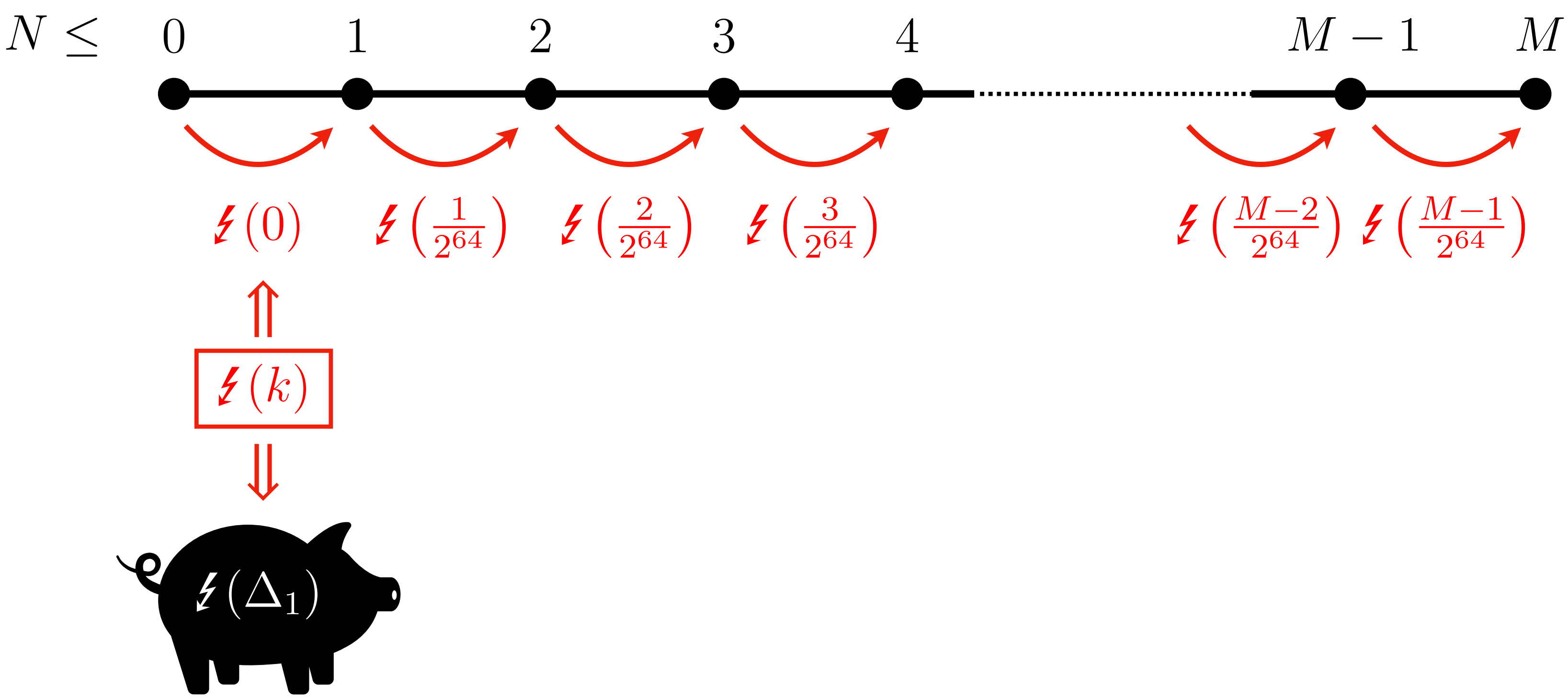
Amortize over M hashes

Property: collisionFree N

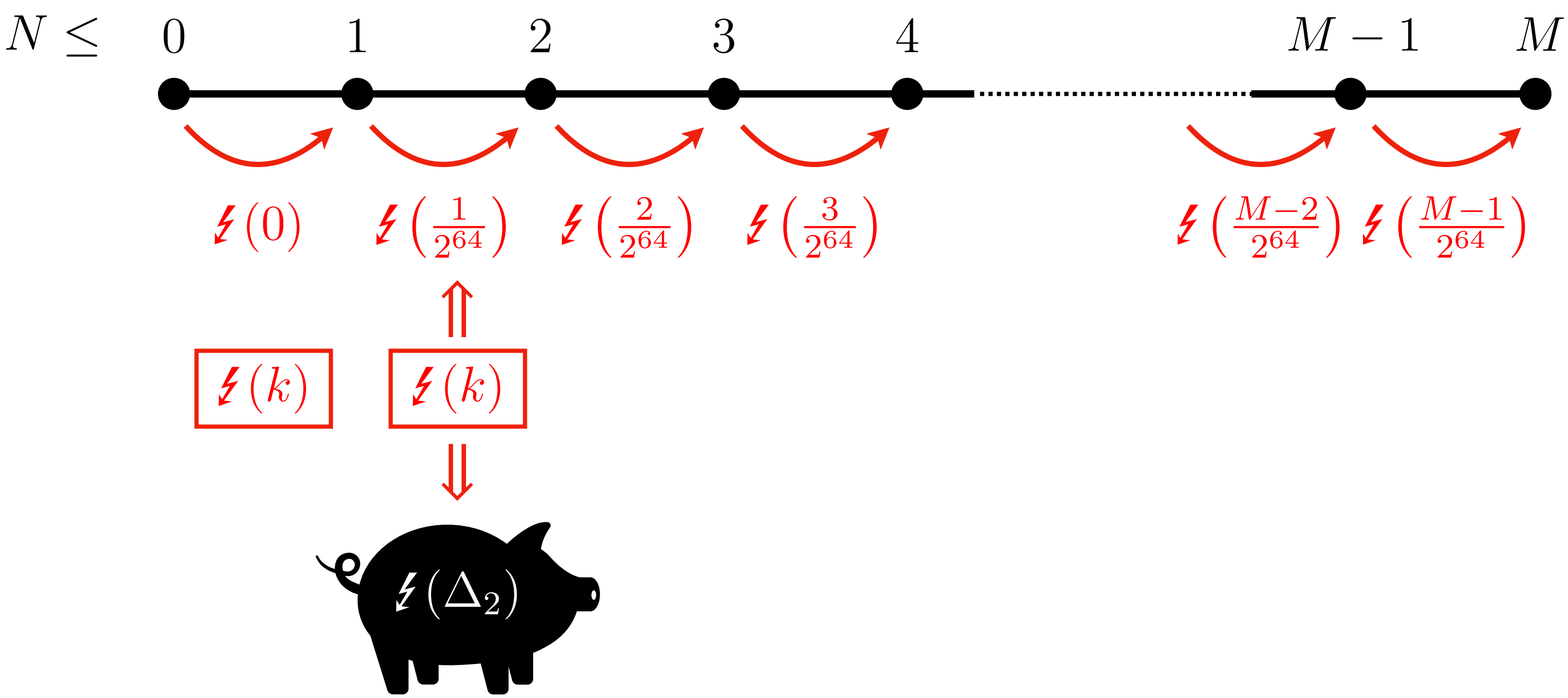
Amortized Credit Arithmetic



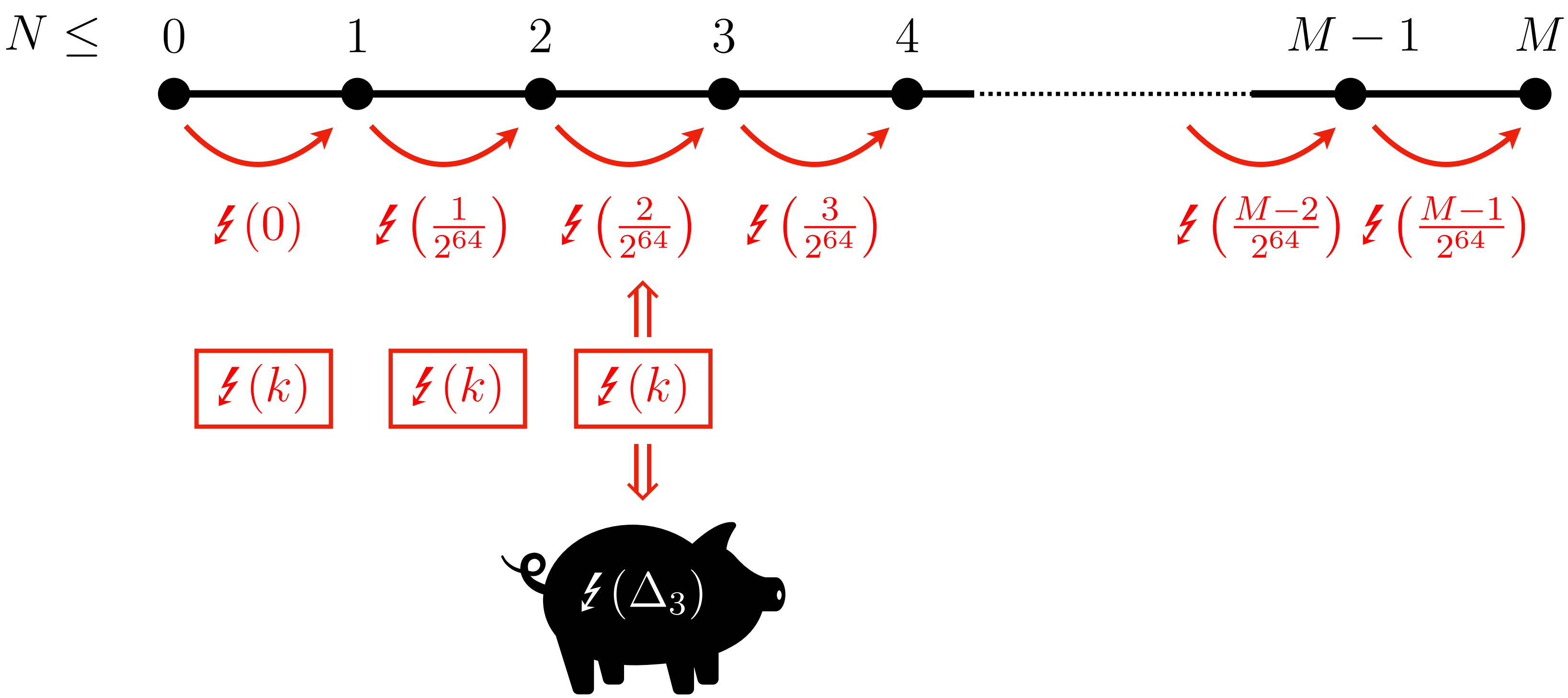
Amortized Credit Arithmetic



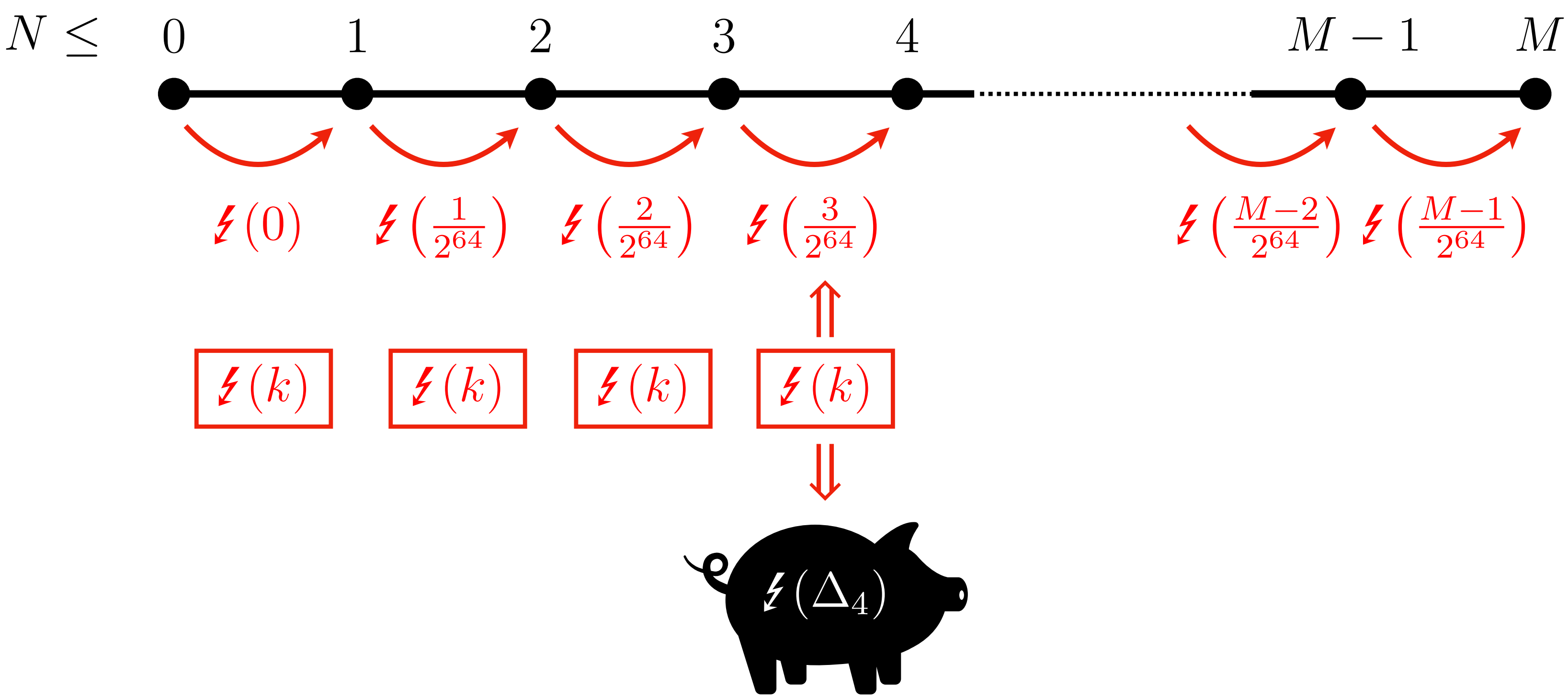
Amortized Credit Arithmetic



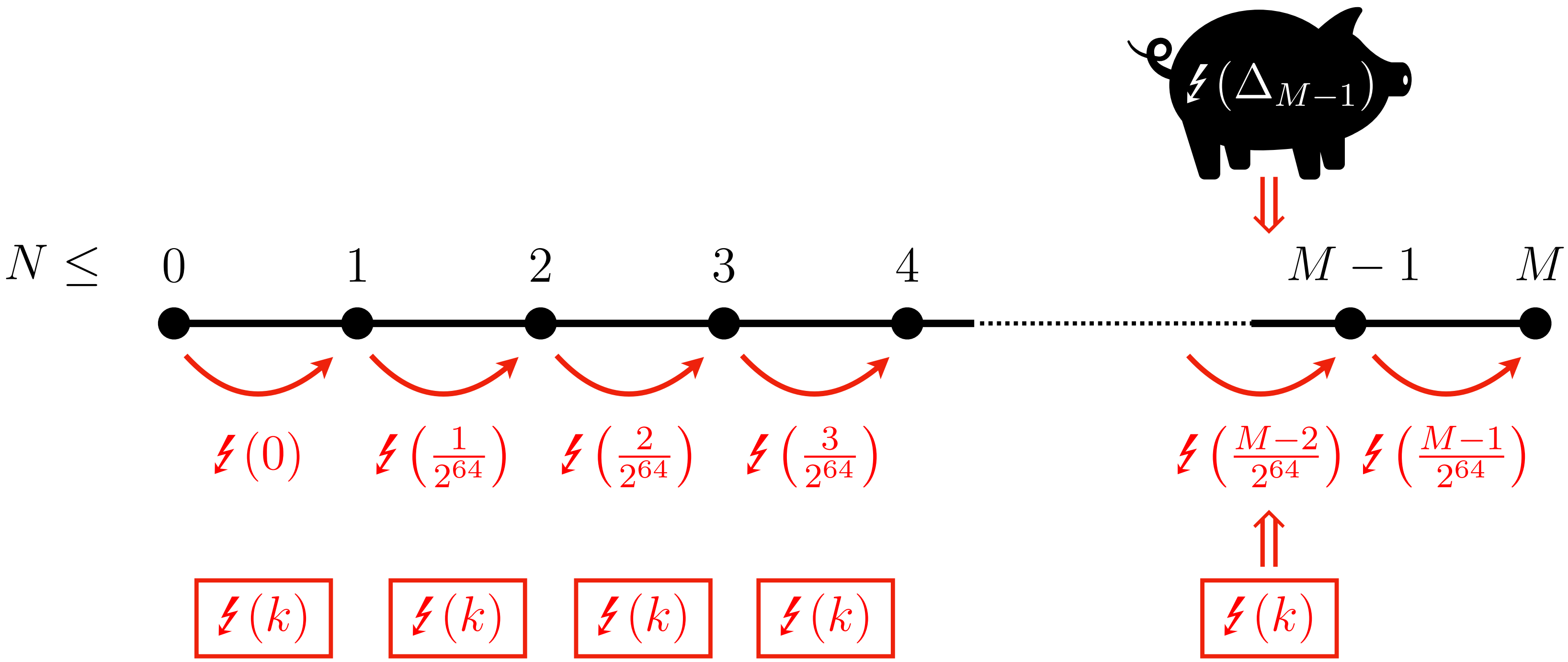
Amortized Credit Arithmetic



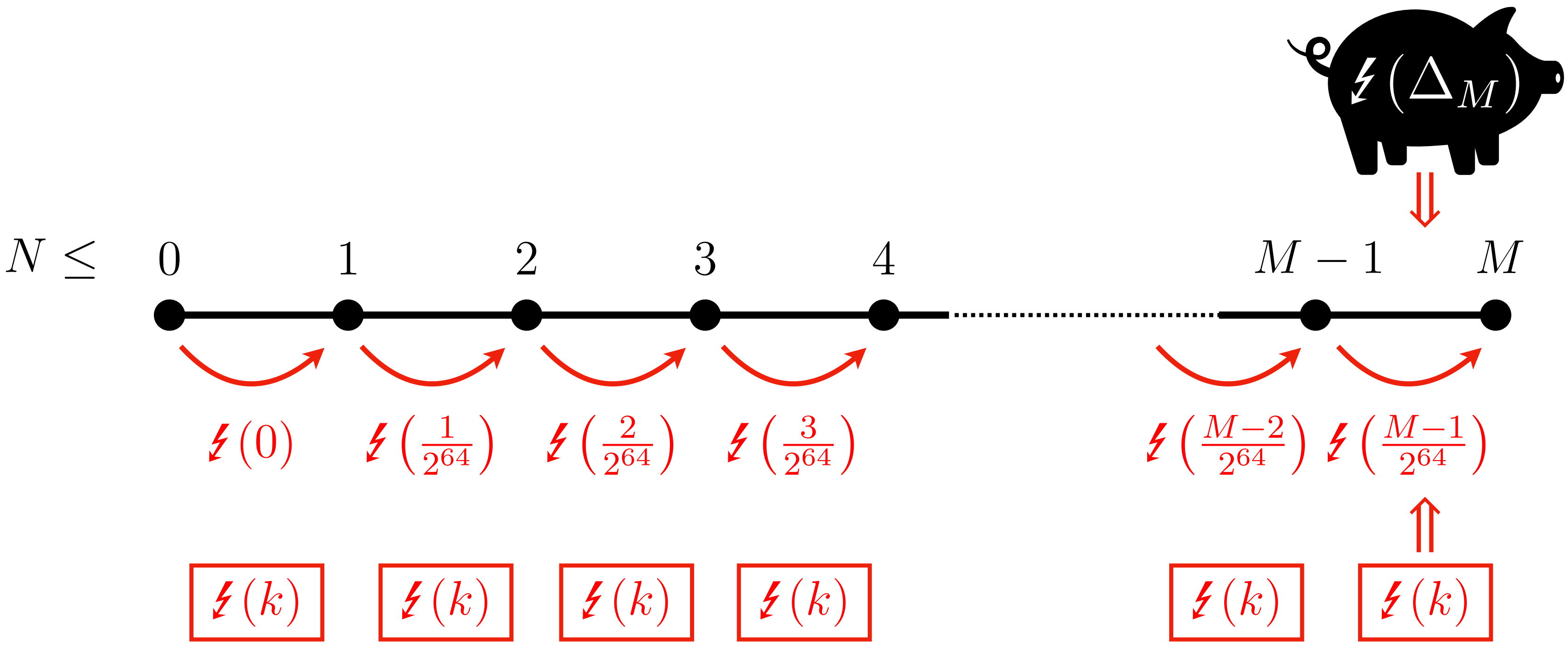
Amortized Credit Arithmetic



Amortized Credit Arithmetic



Amortized Credit Arithmetic



Hash Collisions

hash : $A \to$ int64

```
hash x =  match get x with
            Some (v) ⇒ v
          | None ⇒  let v = sample(2^64) in
                    set x v;
                    v
          end
```

$$\left\{ \begin{array}{c} \text{collisionFree } N \ * \\ ⚡(N \cdot 2^{-64}) \end{array} \right\} \text{hash } x \left\{ v. \begin{array}{c} \text{collisionFree } (N+1) \ * \\ \text{get } x = v \end{array} \right\}$$

Simplify client dependency on N?

Property: collisionFree N

Hash Collisions

hash : $A \rightarrow$ int64

hash $x =$ match get x with
  Some $(v) \Rightarrow v$
  | None $\Rightarrow$ let $v =$ sample(2^{64}) in
    set x v;
    v
  end

$$\left\{ \begin{array}{c} \text{collisionFree } N * \\ \lightning(N \cdot 2^{-64}) \end{array} \right\} \text{hash } x \left\{ v. \begin{array}{c} \text{collisionFree } (N+1) * \\ \text{get } x = v \end{array} \right\}$$

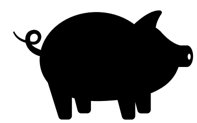
Property: collisionFree N

Hash Collisions

hash : $A \rightarrow$ int64

$$
\begin{aligned}
\text{hash } x = \; & \text{match get } x \text{ with} \\
& \quad \text{Some } (v) \Rightarrow v \\
& \mid \text{None} \Rightarrow \text{let } v = \text{sample}(2^{64}) \text{ in} \\
& \qquad\qquad \text{set } x\ v; \\
& \qquad\qquad v \\
& \text{end}
\end{aligned}
$$

$$\left\{ \begin{array}{c} \text{collisionFree } N * \\ I(N) * N < M * \lightning(k) \end{array} \right\} \text{hash } x \left\{ \begin{array}{c} \text{collisionFree } (N+1) * \\ I(N+1) \end{array} \right\}$$



$$I(N) \triangleq (N \leq M) * \lightning(\Delta_N)$$

$$\left\{ \begin{array}{c} \text{collisionFree } N * \\ \lightning(N \cdot 2^{-64}) \end{array} \right\} \text{hash } x \left\{ v. \begin{array}{c} \text{collisionFree } (N+1) * \\ \text{get } x = v \end{array} \right\}$$

Property: collisionFree N

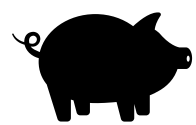
Hash Collisions

hash : $A \to$ int64

hash $x =$ match get x with
  Some $(v) \Rightarrow v$
  | None $\Rightarrow$ let $v =$ sample(2^{64}) in
    set x v;
    v
  end

Derived!

$$\left\{ \begin{array}{c} \text{collisionFree } N \ast \\ I(N) \ast N < M \ast \lightning(k) \end{array} \right\} \text{hash } x \left\{ \begin{array}{c} \text{collisionFree } (N+1) \ast \\ I(N+1) \end{array} \right\}$$

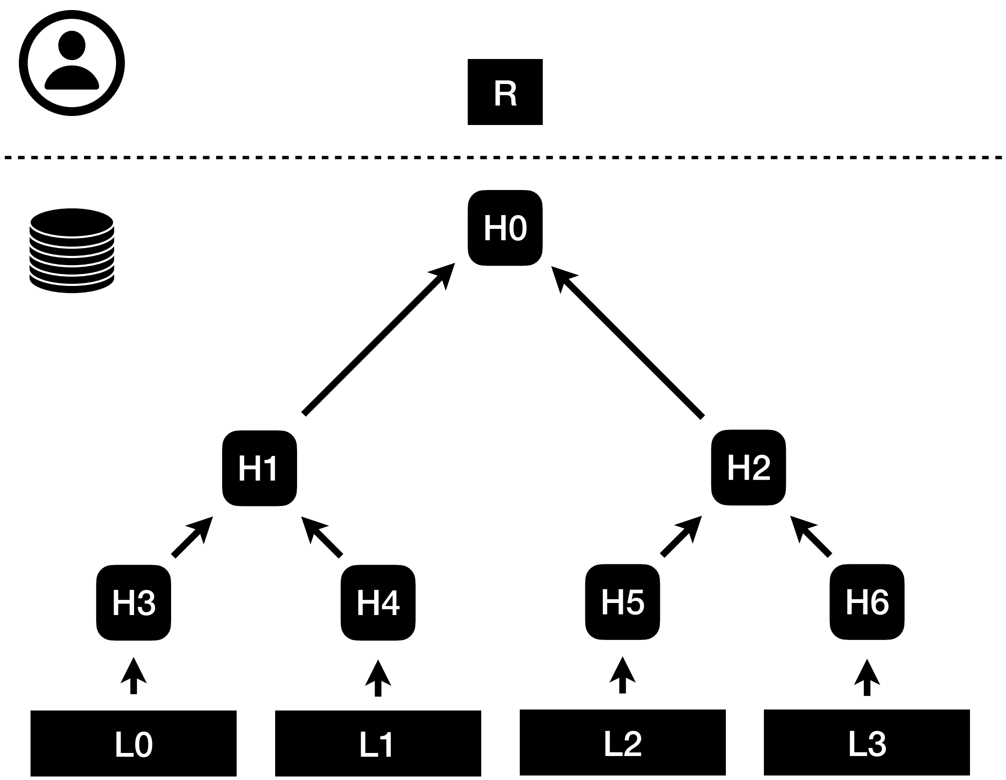


$$I(N) \triangleq (N \leq M) \ast \lightning(\Delta_N)$$

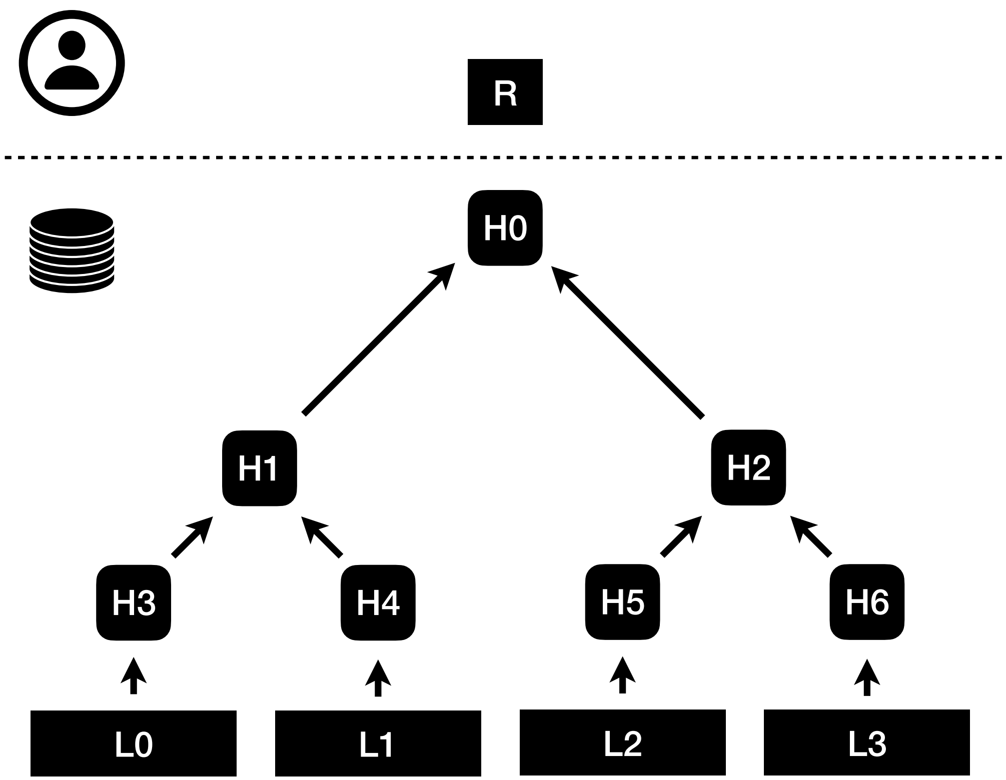
$$\left\{ \begin{array}{c} \text{collisionFree } N \ast \\ \lightning(N \cdot 2^{-64}) \end{array} \right\} \text{hash } x \left\{ v. \begin{array}{c} \text{collisionFree } (N+1) \ast \\ \text{get } x = v \end{array} \right\}$$

Property: collisionFree N

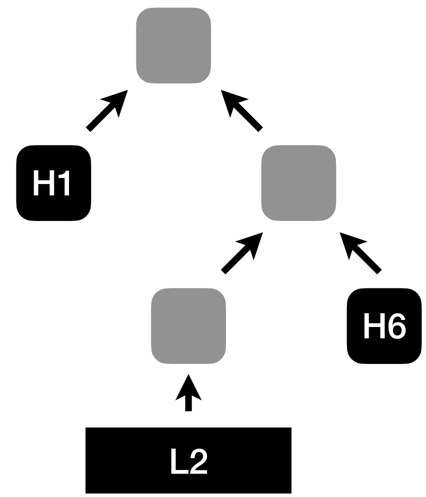
Merkle Tree



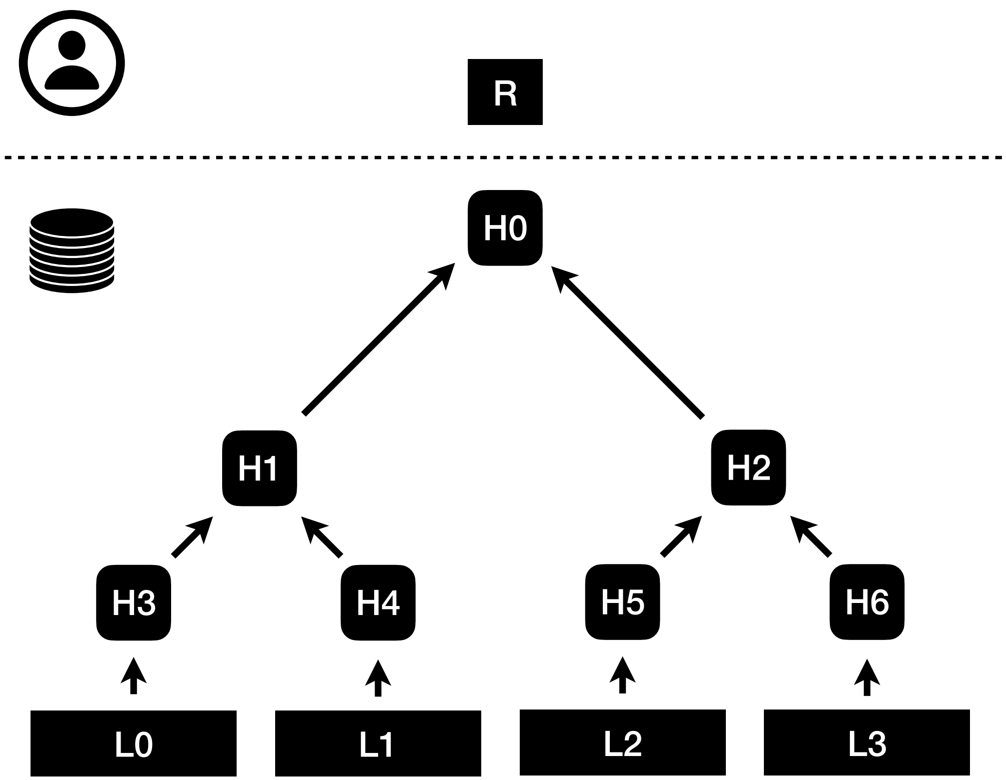
Merkle Tree



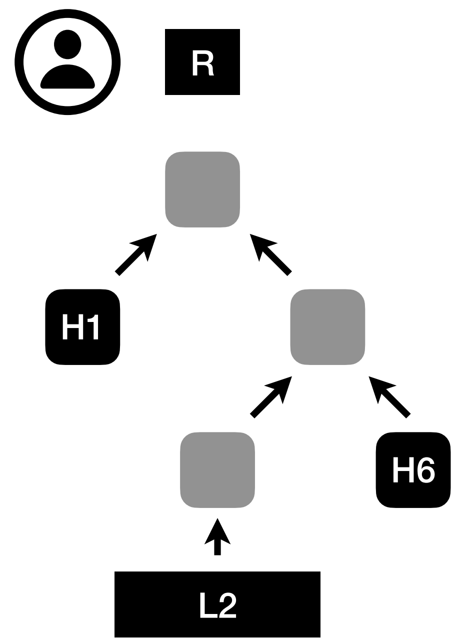
query(L2) =



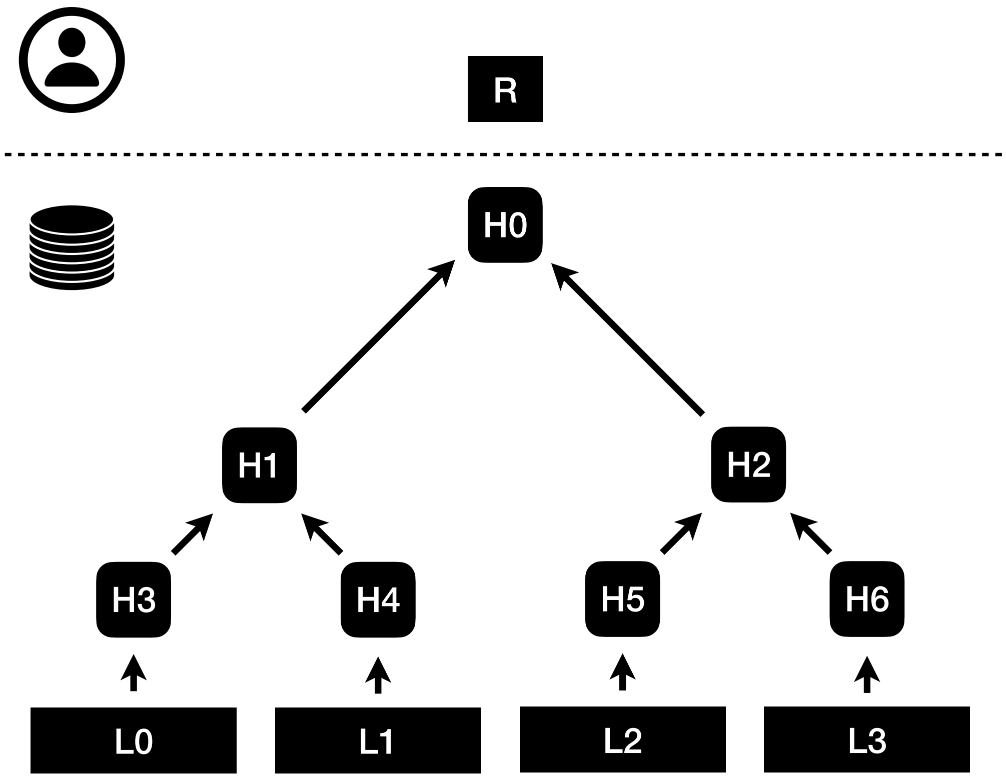
Merkle Tree



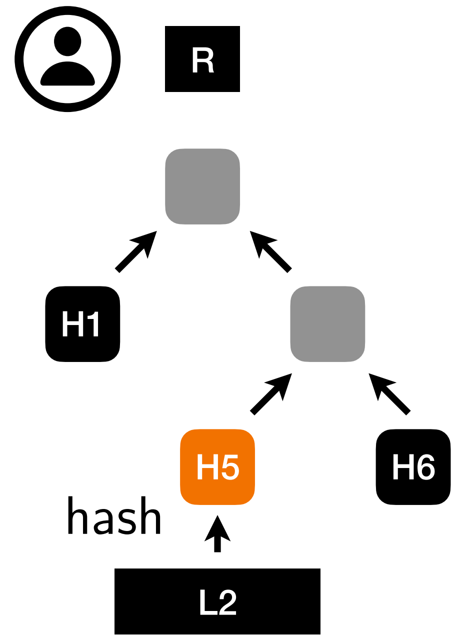
query(L2) =



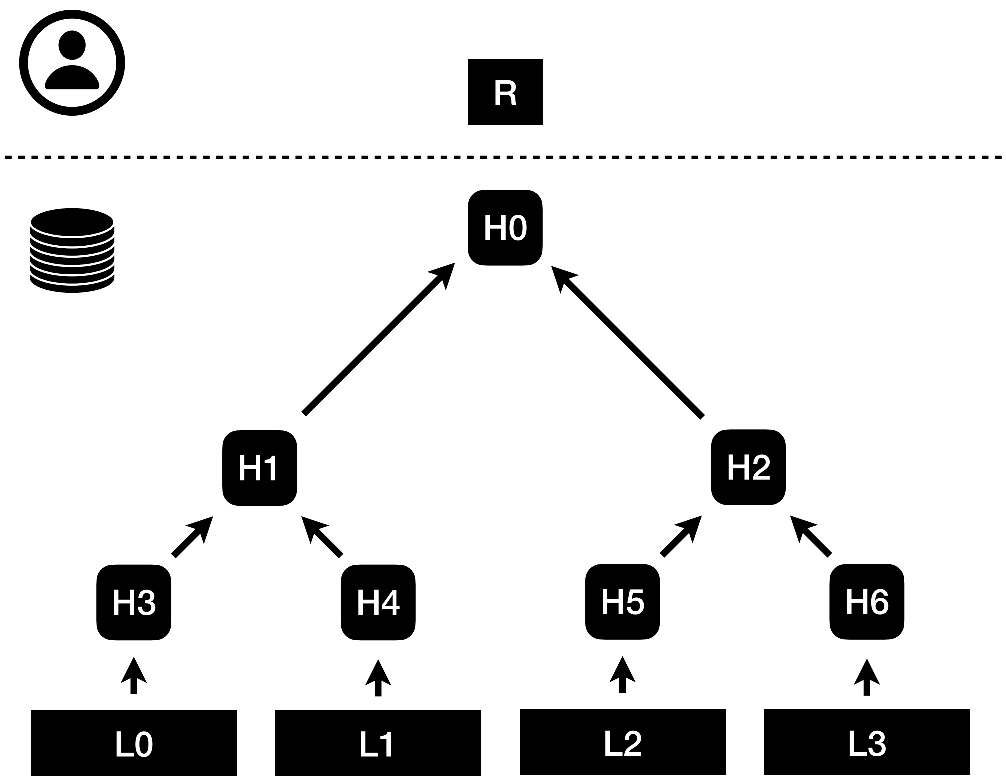
Merkle Tree



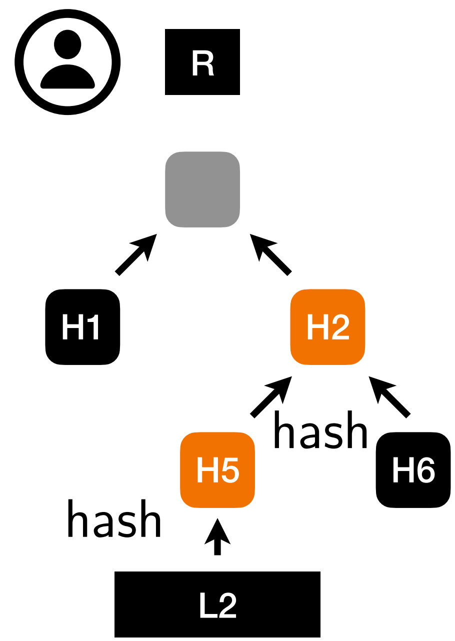
query(L2) =



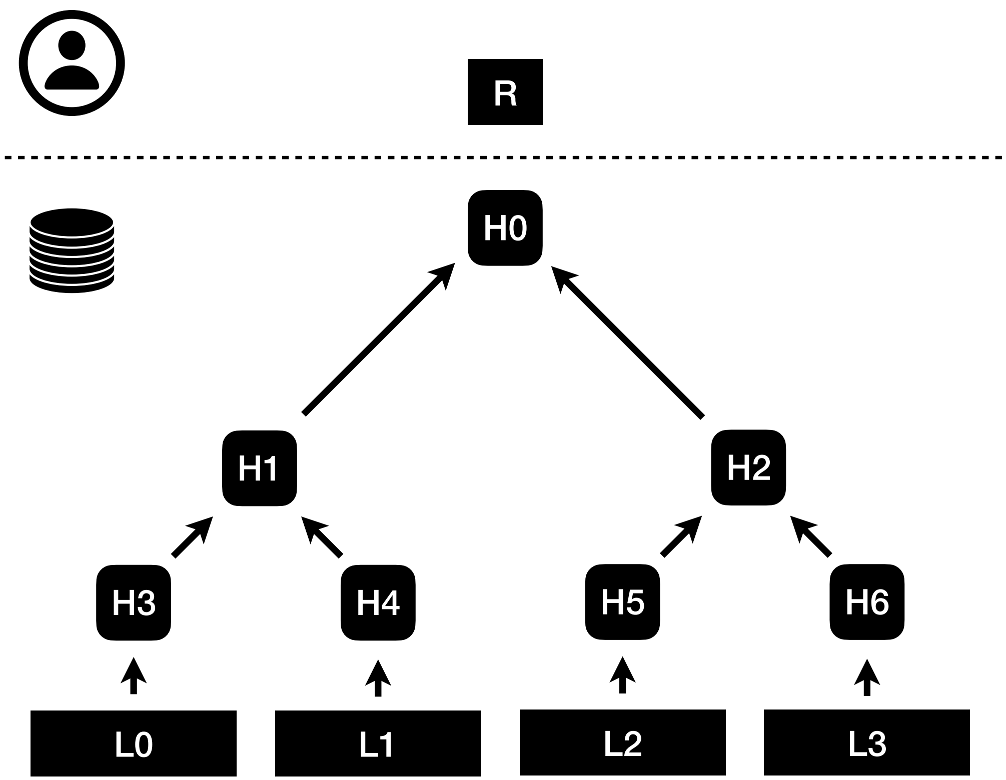
Merkle Tree



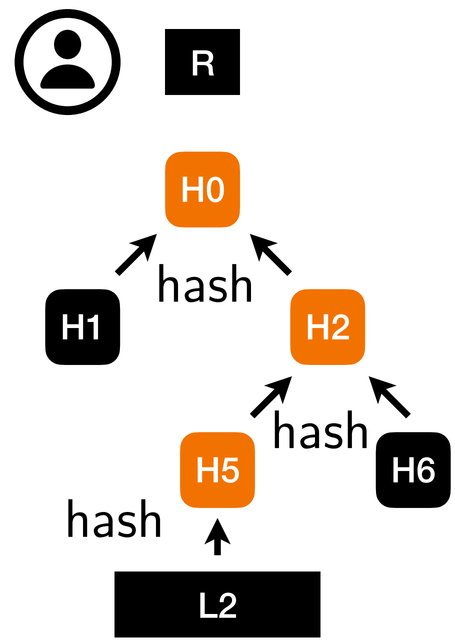
query(L2) =



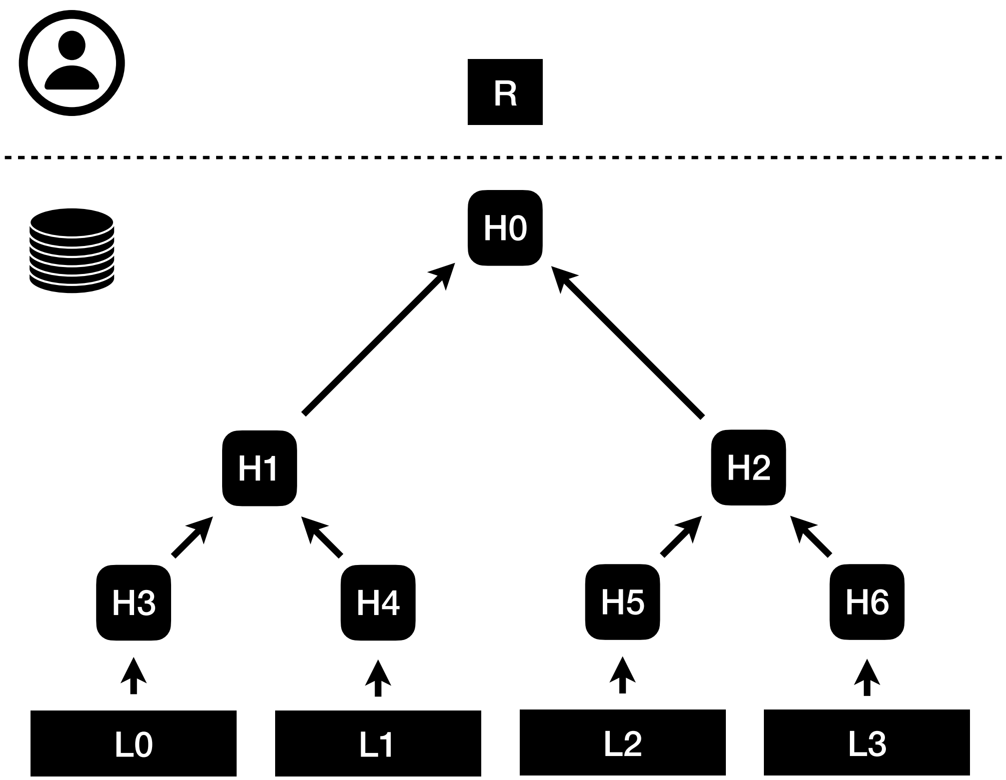
Merkle Tree



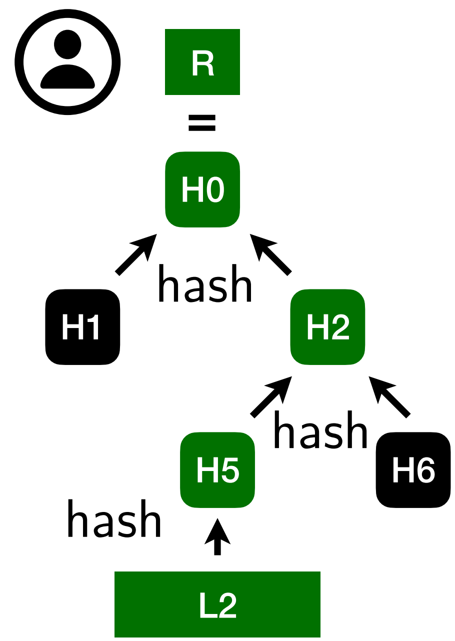
query(L2) =



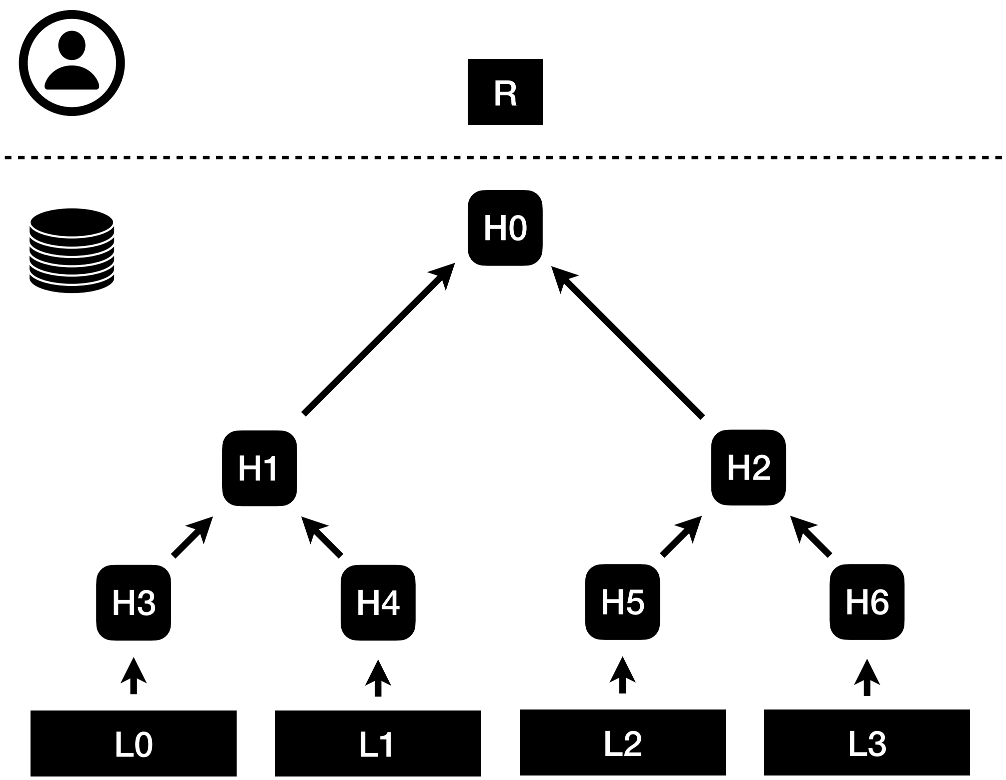
Merkle Tree



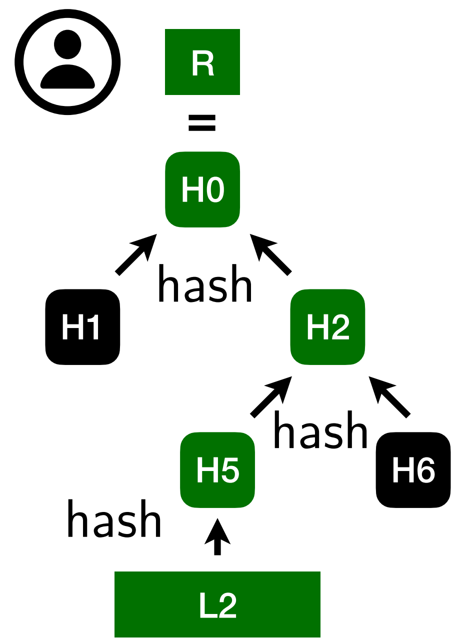
query(L2) =



Merkle Tree



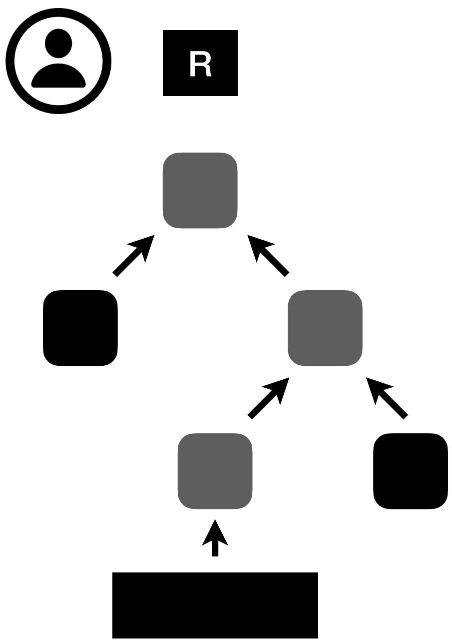
query(L2) =



What are the chances that randomly corrupted data will pass this check?

Merkle Tree

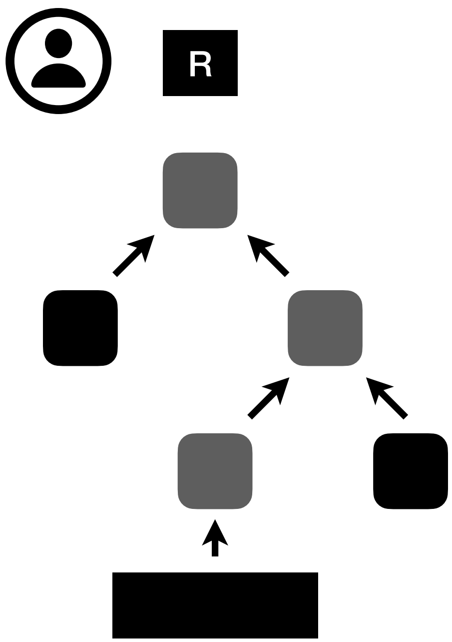
What are the chances that randomly corrupted data will pass this check?



- **Validation program** check

Merkle Tree

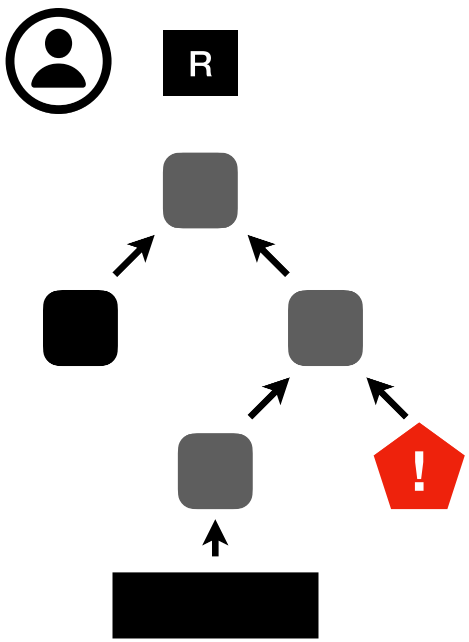
What are the chances that randomly corrupted data will pass this check?



- **Validation program** check
- **Collision free** $\Rightarrow$ check **is sound**

Merkle Tree

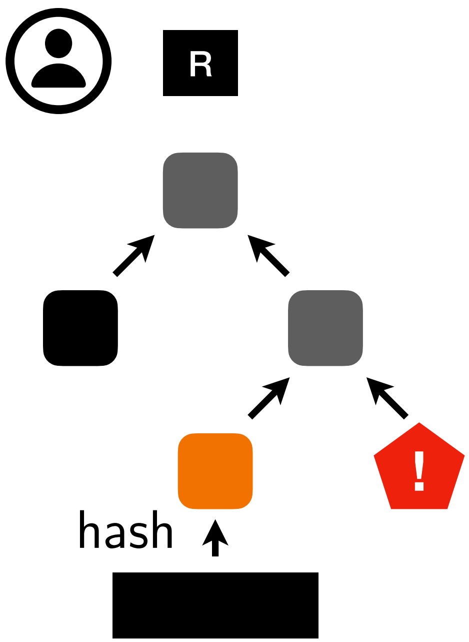
What are the chances that randomly corrupted data will pass this check?



- **Validation program** check
- **Collision free ⇒** check **is sound**

Merkle Tree

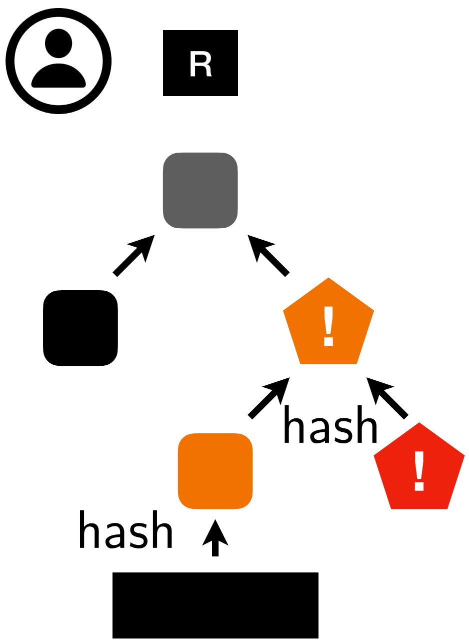
What are the chances that randomly corrupted data will pass this check?



- **Validation program** check
- **Collision free** $\Rightarrow$ check **is sound**

Merkle Tree

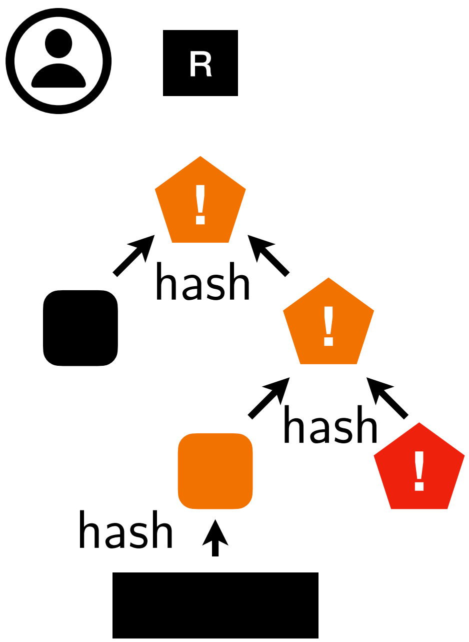
What are the chances that randomly corrupted data will pass this check?



- **Validation program** check
- **Collision free** ⇒ check **is sound**

Merkle Tree

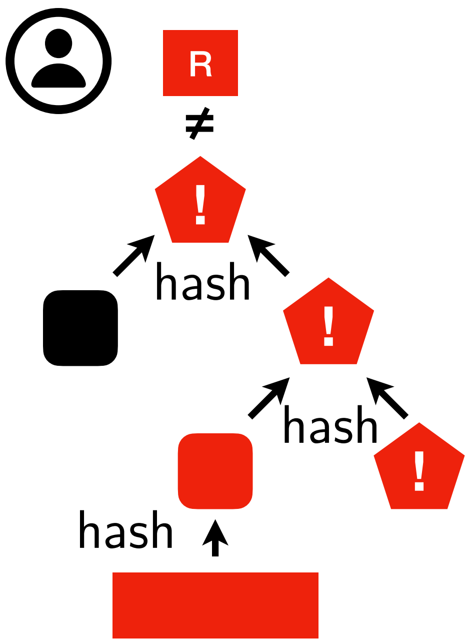
What are the chances that randomly corrupted data will pass this check?



- **Validation program** check
- **Collision free** ⇒ check **is sound**

Merkle Tree

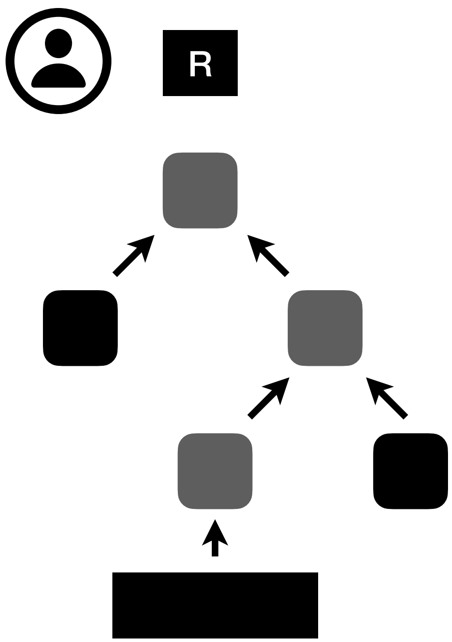
What are the chances that randomly corrupted data will pass this check?



- **Validation program** check
- **Collision free** ⇒ check **is sound**

Merkle Tree

What are the chances that randomly corrupted data will pass this check?

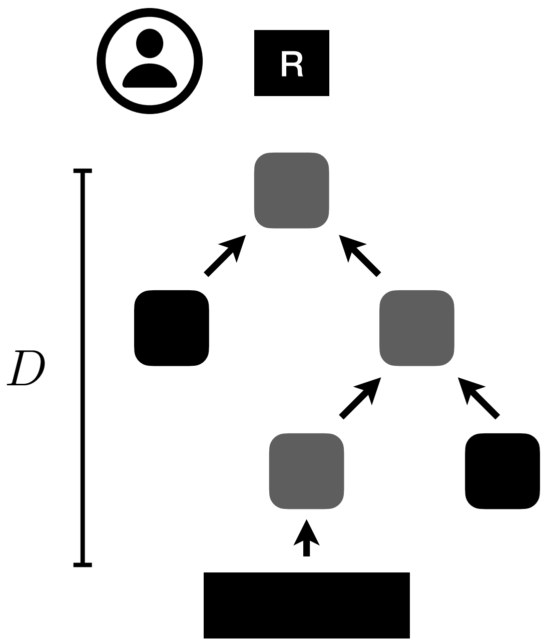


- **Validation program** check
- **Collision free ⇒** check **is sound**

$$\left\{ \begin{array}{c} \text{collisionFree } N \, * \\ I(N) * N < M * ↯(k) \end{array} \right\} \text{hash } x \left\{ \begin{array}{c} \text{collisionFree } (N+1) \, * \\ I(N+1) \end{array} \right\}$$

Merkle Tree

What are the chances that randomly corrupted data will pass this check?



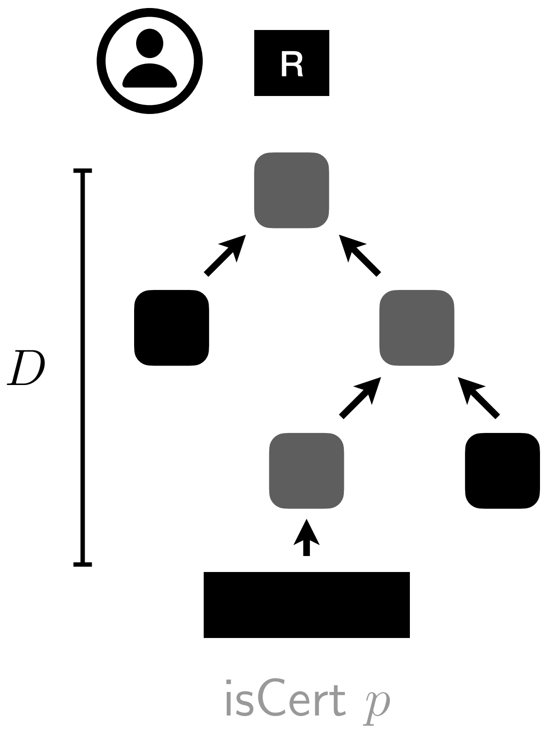
isCert p

- **Validation program** check
- **Collision free** ⇒ check **is sound**

$$\left\{ \begin{array}{c} \text{collisionFree } N \ast \\ I(N) \ast N < M \ast \lightning(k) \end{array} \right\} \text{hash } x \left\{ \begin{array}{c} \text{collisionFree } (N+1) \ast \\ I(N+1) \end{array} \right\}$$

Merkle Tree

What are the chances that randomly corrupted data will pass this check?



- **Validation program** check
- **Collision free** $\Rightarrow$ check **is sound**

$$\left\{ \begin{array}{c} \text{collisionFree } N * \\ I(N) * N < M * \lightning(k) \end{array} \right\} \text{hash } x \left\{ \begin{array}{c} \text{collisionFree } (N+1) * \\ I(N+1) \end{array} \right\}$$

$$\left\{ \begin{array}{c} \text{collisionFree } N * I(N) * \\ N + D < M * \text{isCert } p * \\ \lightning(k \cdot D) \end{array} \right\} \text{check } p \left\{ \begin{array}{c} \text{collisionFree } (N+D) * \\ I(N+D) \end{array} \right\}$$

At most $\lightning(k \cdot D)$

Current and Future Work

coauthors at NESVD

- Expected termination bounds
- Randomized SAT solver
- Rejection samplers
- Resizing Hash Tables

Simon

Joe

Markus

Thank you for your attention!