Tallinn 19 March 2024

EFFECTFUL TRACE SEMANTICS VIA EFFECTFUL STREAMS

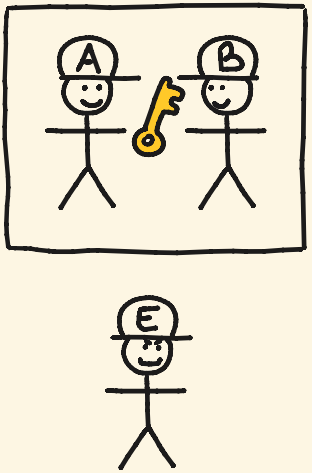
Filippo Bonchi
Università di Pisa

Elena Di Lavore
Università di Pisa

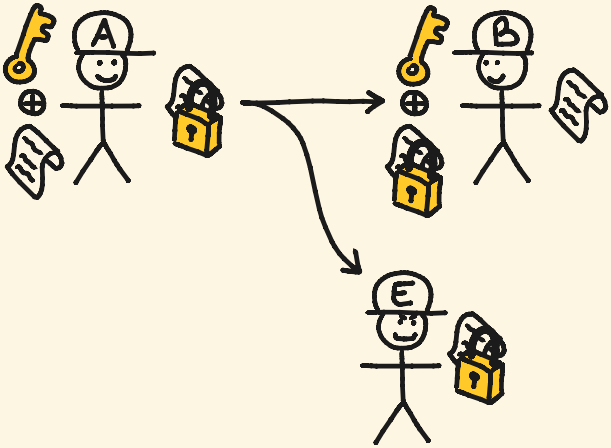
Mario Román
University of Oxford

ONE-TIME PAD PROTOCOL

1. share a key through a secure channel



2. send an encrypted message through a public channel



[Broadbent & Karvonen 2023]

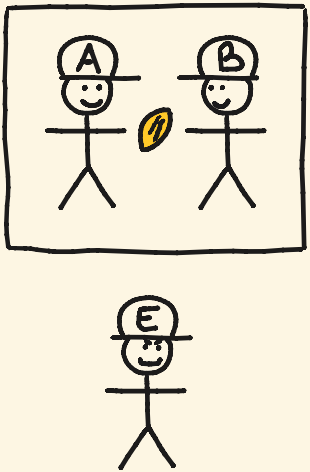
REPEATING THE ONE-TIME PAD

Sending n messages securely requires n private keys
↳ not very useful

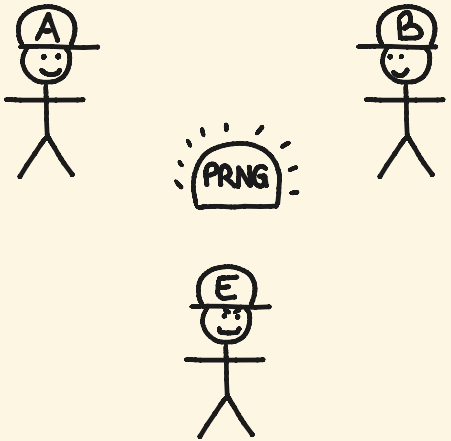
⇒ • privately share a seed
• use identical pseudorandom number generators to obtain a new key for each message

STREAM CIPHER PROTOCOL (1)

1. share a seed through a secure channel

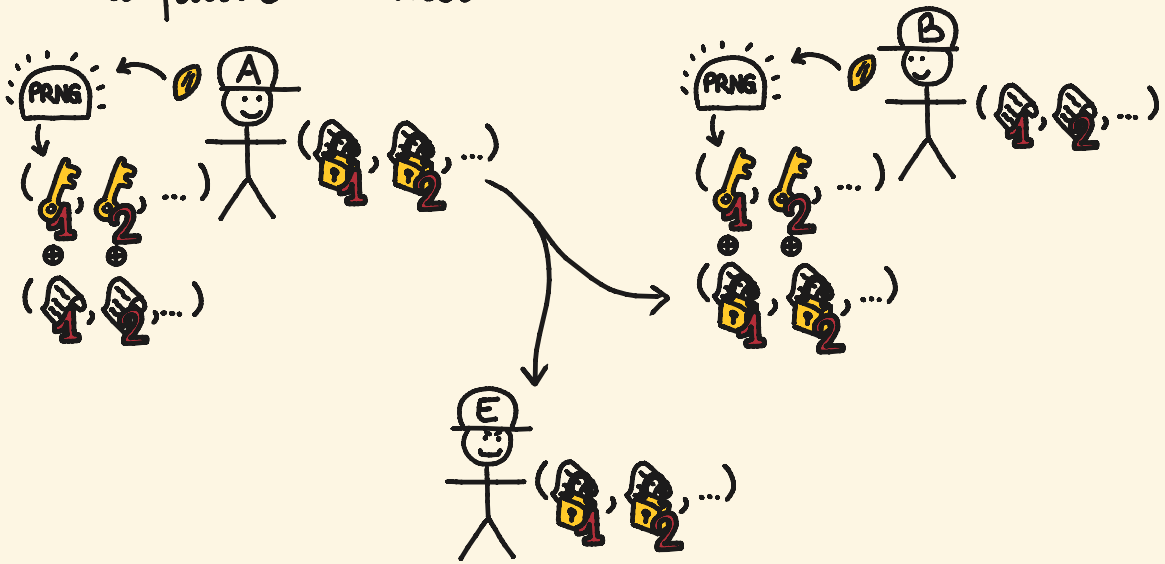


2. share a pseudorandom number generator



STREAM CIPHER PROTOCOL (2)

3. send a stream of encrypted messages through a public channel



STREAM CIPHER PROTOCOL COINDUCTIVELY

$$\begin{aligned}
&\mathsf{seedGen}()^{\circ} = \mathbf{do}\\
&\quad \mathsf{rand}() \to s\\
&\quad \mathsf{setSeed}(s) \rightsquigarrow ()\\
&\quad \mathbf{return}\ ()
\end{aligned}$$

$$\begin{aligned}
&\mathsf{seedGen}()^{+\circ} = \mathbf{do}\\
&\quad \mathbf{return}\ ()
\end{aligned}$$

$$\mathsf{seedGen}^{++} = \mathsf{seedGen}^{+}$$

$$\begin{aligned}
&\mathsf{alice}(m)^{\circ} = \mathbf{do}\\
&\quad \mathsf{getSeedA}() \rightsquigarrow (s)\\
&\quad \mathsf{prng}(s) \to (s', k)\\
&\quad \mathbf{return}\ (s', m \oplus k)
\end{aligned}$$

$$\begin{aligned}
&\mathsf{alice}(s, m)^{+\circ} = \mathbf{do}\\
&\quad \mathsf{prng}(s) \to (s', k)\\
&\quad \mathbf{return}\ (s', m \oplus k)
\end{aligned}$$

$$\mathsf{alice}(s, m)^{++} = \mathsf{alice}(s, m)^{+}$$

STREAM CIPHER PROTOCOL COINDUCTIVELY

$$\begin{aligned}
&bob(m)^{\circ} = \mathbf{do}\\
&\quad getSeedB() \rightsquigarrow (s)\\
&\quad prng(s) \to (s', k)\\
&\quad \mathbf{return}\ (s', m \oplus k)
\end{aligned}$$

$$\begin{aligned}
&bob(s, m)^{+\circ} = \mathbf{do}\\
&\quad prng(s) \to (s', k)\\
&\quad \mathbf{return}\ (s', m \oplus k)
\end{aligned}$$

$$bob(s, m)^{++} = bob(s, m)^{+}$$

$$\begin{aligned}
&eve(m)^{\circ} = \mathbf{do}\\
&\quad \mathbf{return}\ (m)
\end{aligned}$$

$$eve(m)^{+} = eve(m)$$

OUTLINE

- effectful categories
- effectful streams
- effectful trace semantics
- causal processes

COMPUTATIONS WITH EFFECTS

- Stochastic effects: the distribution monad

  $\mathcal{D} : \mathbf{Set} \to \mathbf{Set}$

  $\mathcal{D}(A) := \{ \sigma : A \to [0,1] \mid \mathrm{supp}\, \sigma \text{ is finite} \wedge \sum_{a \in A} \sigma(a) = 1 \}$

- Global state: the state promonad

  $\mathbf{St} : \mathcal{C}^{op} \times \mathcal{C} \to \mathbf{Set}$

  $\mathbf{St}(A,B) := \mathcal{C}(S \otimes A, S \otimes B)$

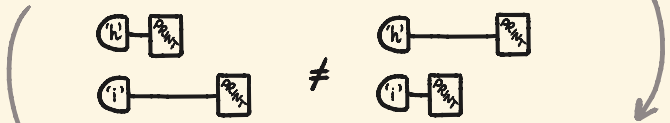
PREMONOIDAL CATEGORIES

Some effects do not interchange.

```
printHI() = do
  print('h') ⇝ ()
  print('i') ⇝ ()
  return ()
```

≠

```
printIH() = do
  print('i') ⇝ ()
  print('h') ⇝ ()
  return ()
```



prints "hi" prints "ih"

ex state promonads, IO monad

STREAM CIPHER PROTOCOL (AGAIN)

$\mathsf{seedGen}()^{\circ} =$ do
- $\mathsf{rand}() \to s$
- $\mathsf{setSeed}(s) \rightsquigarrow ()$
- return $()$

$\mathsf{seedGen}()^{+\circ} =$ do
- return $()$

$\mathsf{seedGen}^{++} = \mathsf{seedGen}^{+}$

$\mathsf{alice}(m)^{\circ} =$ do
- $\mathsf{getSeedA}() \rightsquigarrow (s)$
- $\mathsf{prng}(s) \to (s', k)$
- return $(s', m \oplus k)$

$\mathsf{alice}(s, m)^{+\circ} =$ do
- $\mathsf{prng}(s) \to (s', k)$
- return $(s', m \oplus k)$

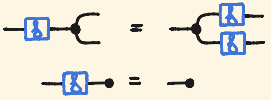
$\mathsf{alice}(s, m)^{++} = \mathsf{alice}(s, m)^{+}$

$\mathsf{eve}(m)^{\circ} =$ do
- return (m)

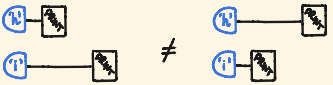
$\mathsf{eve}(m)^{+} = \mathsf{eve}(m)$

EFFECTFUL COPY-DISCARD CATEGORIES

Values can be copied and discarded (cartesian)



Effectful computations may have global effects (premonoidal)



$\mathcal{V} \hookrightarrow \mathcal{L} \hookrightarrow \mathcal{C}$

Local computations interchange (monoidal)

ex (Set, Stoch, State)

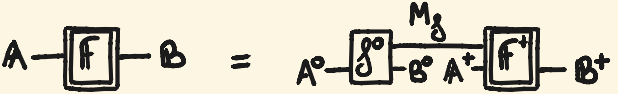
OUTLINE

- effectful categories
- effectful streams
- effectful trace semantics
- causal processes

EFFECTFUL STREAMS

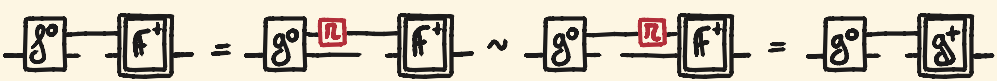
An effectful stream $F : A \to B$ on $(\mathcal{V}, \mathcal{L}, \mathcal{C})$ is

- a memory $M_g \in \mathcal{L}$
- a first action $g^\circ : A^\circ \to M_g \otimes B^\circ$ in $\mathcal{C}$
- the rest of the action $F^+ : M_g \cdot A^+ \to B^+$



quotiented by the equivalence relation generated by

$$\begin{cases} g^\circ ; (\pi \otimes 1) = g^\circ \\ F^+ = \pi \cdot g^+ \end{cases} \qquad \text{for } \pi : M_g \to M_g \text{ in } \mathcal{L}$$



COMPOSITIONAL STRUCTURE OF STREAMS

THEOREM

Effectful streams form an effectful category Stream.

- composition and monoidal actions are defined coinductively: for $f : N_g \cdot A \to B$ and $g : N_g \cdot B \to C$,

$$\begin{cases} (f \fatsemi_N g)^\circ := \\ (f \fatsemi_N g)^+ := f^+ \fatsemi_M g^+ \end{cases}$$

$$\begin{cases} (X \otimes_N f)^\circ := \\ (X \otimes_N f)^+ := X^+ \otimes_M f^+ \end{cases}$$

SEMANTICS FOR THE STREAM CIPHER PROTOCOL

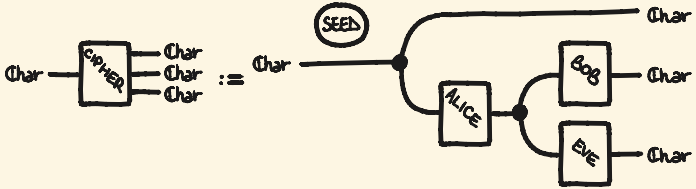
Fix two finite sets Char, Seed
and take the effectful copy-discard category

$$(\mathbf{Set}, \mathbf{Stoch}, \mathbf{SeedStoch})$$

SeedStoch is the Kleisli category of the promonad
that adds the global state $\mathrm{Seed} \times \mathrm{Seed}$:

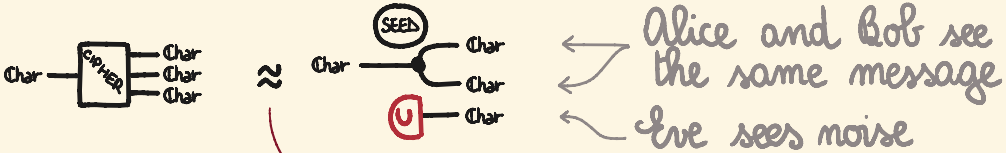
$$\mathbf{SeedStoch}(A, B) := \mathbf{Stoch}(\mathrm{Seed} \times \mathrm{Seed} \times A, \mathrm{Seed} \times \mathrm{Seed} \times B)$$

SEMANTICS FOR THE STREAM CIPHER PROTOCOL



THEOREM

The stream cipher protocol is secure.



[cf. Broadbent & Karvonen 2023]

OUTLINE

- effectful categories
- effectful streams
- [effectful trace semantics]
- causal processes

EFFECTFUL MEALY MACHINES

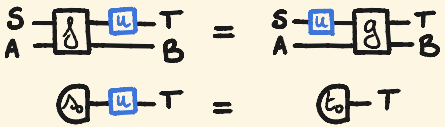
A Mealy machine $(f, S, s_0) : A \to B$ in $(\mathcal{V}, \mathcal{L}, \mathcal{C})$ is a morphism

$$f : S \otimes A \to S \otimes B$$

with an initial state

$$s_0 : I \to S$$

A morphism of Mealy machines $u : (f, S, s_0) \to (g, T, t_0)$ is a value morphism $u : S \to T$ in $\mathcal{V}$ such that

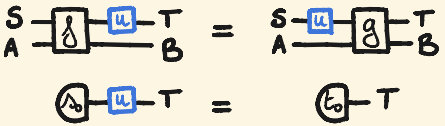


[cf. Katis, Sabadini, Walters 1997 ; EDL, Gianola, Román, Sabadini, Sobociński 2022]

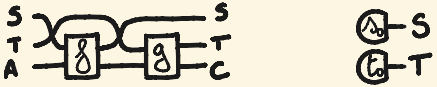
EFFECTFUL CATEGORY OF MEALY MACHINES

Mealy is an effectful category where

- objects are the objects of $\mathcal{C}$
- morphisms $(f, S, s_0) : A \to B$ are Mealy machines quotiented by value isomorphisms $u : S \xrightarrow{\cong} T$



- composition tensors the state spaces



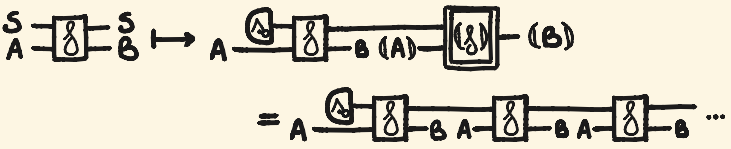
COMPOSITIONAL TRACE SEMANTICS

THEOREM

There is an effectful functor

$$\mathrm{Tr} : \mathbf{Mealy} \to \mathbf{Stream}$$

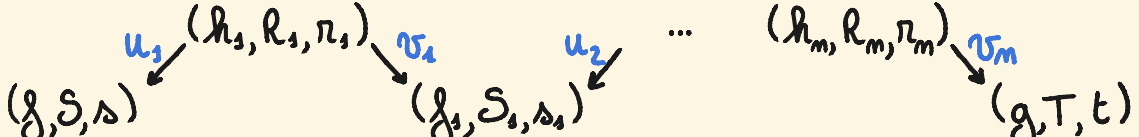
$$A \mapsto (\!|A|\!) = (A, A, \dots)$$



$\rightsquigarrow$ in **Rel** these traces coincide with the classical traces

BISIMULATION

Two effectful Mealy machines $(g, S, s), (g, T, t) : A \to B$ are <u>bisimilar</u> if they belong to the same connected component in $\mathrm{Mealy}(A, B)$:



THEOREM

For Mealy machines in $(\mathcal{V}, \mathcal{L}, \mathcal{C})$, bisimulation implies trace equivalence.

PROOF: By coinduction. □

COALGEBRAIC BISIMULATION

PROPOSITION

When $\mathcal{C} = \mathcal{K}\ell(M)$, for a commutative monad preserving weak pullbacks, then (f,S,s) and (g,T,t) are bisimilar iff they have the same bisimulation quotient, i.e. there is (h,Q,q) with morphisms

$$(f,S,s) \xrightarrow{u} (h,Q,q) \xleftarrow{v} (g,T,t) \ .$$

EXAMPLES

- Set
- Rel
- pStoch

OUTLINE

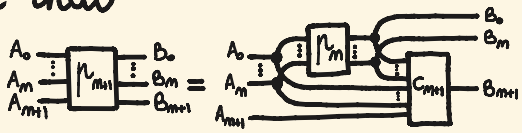
- effectful categories
- effectful streams
- effectful trace semantics
- [causal processes]

CAUSAL PROCESSES

A causal process $p : A \to B$ in a copy-discard category $\mathcal{C}$ is a family of morphisms

$$p_n : A_0 \otimes \cdots \otimes A_n \longrightarrow B_0 \otimes \cdots \otimes B_n$$

such that



for some $c_{n+1} : B_0 \otimes \cdots \otimes B_n \otimes A_0 \otimes \cdots \otimes A_n \otimes A_{n+1} \to B_{n+1}$

THEOREM

causal processes form a monoidal category $\mathbf{Proc}$ when $\mathcal{C}$ has quasi-total conditionals.

[cf. Raney 1958 ; Sprunger & Katsumata 2019]

CAUSAL PROCESSES ARE STREAMS

THEOREM

Consider $(\mathrm{fun}\mathcal{C}, \mathrm{tot}\mathcal{C}, \mathcal{C})$.

If $\mathcal{C}$ has quasi-total conditionals and ranges,

$$\mathrm{Proc} \simeq \mathrm{Stream}\ .$$

EXAMPLES

- Set
- Rel
- pStoch
- Par
- Stoch

TRACES ARE EFFECTFUL TRACES

Compute the traces of a Mealy machine

$$(f, S, s) : A \to B$$

in some known cases.

$(b_0, \dots, b_n)$ is a trace of $(a_0, \dots, a_n)$

Set if $s_0 = s$ and $\forall k \leq n \ (s_{k+1}, b_k) = f(s_k, a_k)$

Rel if $\exists (s_0, \dots, s_{n+1}) \quad s_0 \in s$
and $\forall k \leq n \ (s_{k+1}, b_k) \in f(s_k, a_k)$

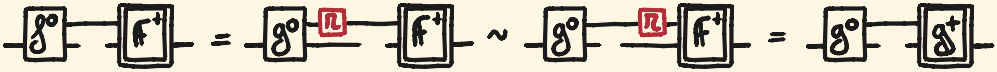
pStoch with probability $\sum_{(s_0, \dots, s_{n+1})} s(s_0 \mid *) \cdot \prod_{k \leq n} f(s_{k+1}, b_k \mid s_k, a_k)$

SUMMARY

- formal compositional semantics for effectful stream computations
- trace equivalence and bisimulation of effectful Mealy machines
- characterisation as causal stream processes

FUTURE WORK

- coinduction up-to dinaturality



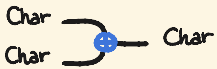
- Rel with explicit failure
- equality in St-C implies bisimulation

- distance instead of equivalence relation for security

SEMANTICS FOR THE STREAM CIPHER PROTOCOL

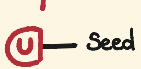
Semantics for values and local computations.

$[\![- \oplus -]\!] := \text{XOR} : \text{Char} \times \text{Char} \to \text{Char}$ in Set



↝ bitwise XOR

$[\![\text{rand}]\!] := \text{unif} : 1 \to \mathcal{D}(\text{Seed})$ in Stoch



↝ uniform distribution

$[\![\text{prng}]\!] : \text{Seed} \to \text{Seed} \times \mathcal{D}(\text{Char})$ in Stoch

↝ use the seed to generate a key

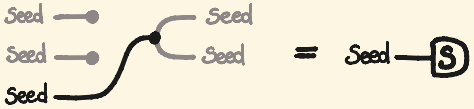
SEMANTICS FOR THE STREAM CIPHER PROTOCOL

Semantics for effectful computations.

$[\![\text{setSeed}]\!] : \text{Seed}^3 \to \text{Seed}^2$

$\text{Seed} \rightsquigarrow 1$

$\rightsquigarrow$ copy the seed to the global state

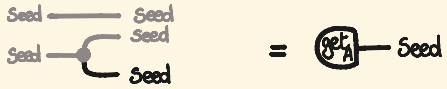
$[\![\text{setSeed}]\!] :=$ 

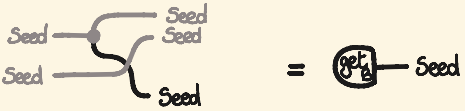
in SeedStoch

$[\![\text{getSeedA}]\!], [\![\text{getSeedB}]\!] : \text{Seed}^2 \to \text{Seed}^3$

$1 \rightsquigarrow \text{Seed}$

$\rightsquigarrow$ alice and bob get their seeds

$[\![\text{getSeedA}]\!] :=$ 

$[\![\text{getSeedB}]\!] :=$ 

SEMANTICS FOR THE STREAM CIPHER PROTOCOL

- seedGen = (SEED) : $\mathbb{I} \to \mathbb{I}$ in Stream

$seedGen^{\circ}$:=

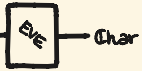
=

$seedGen^{+\circ}$:=

=

$seedGen^{++} = seedGen^{+}$

- eve =



: Char → Char in Stream

eve° :=

=

$eve^{+} = eve$

SEMANTICS FOR THE STREAM CIPHER PROTOCOL

- alice = 

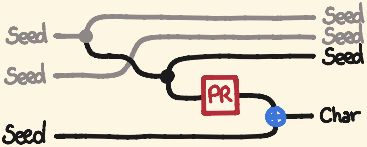
: Char → Char in Stream

- bob = 

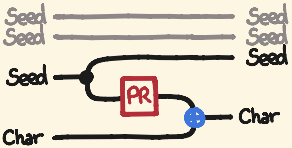
: Char → Char in Stream

alice° := 

= 

bob° := 

= 

alice⁺° = bob⁺° := 

= 

alice⁺⁺ = alice⁺

bob⁺⁺ = bob⁺

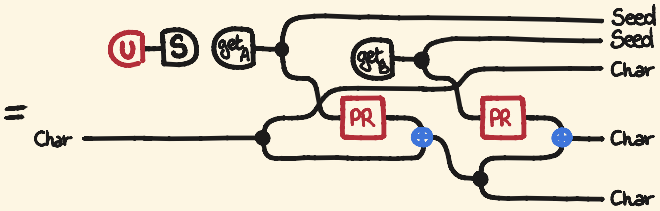
STREAM CIPHER IS SECURE

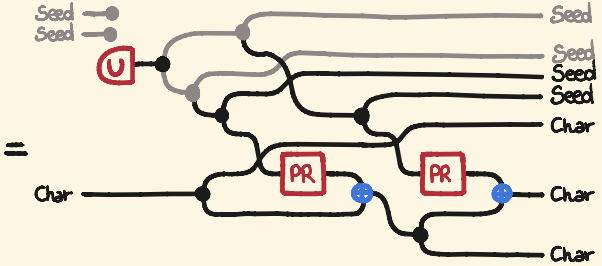
Proceed by coinduction.

cipher°

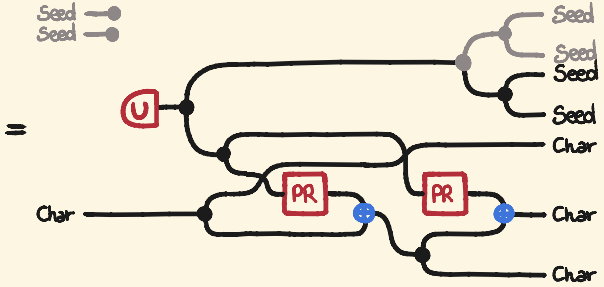
$$
\begin{aligned}
&\mathsf{cipher}(m)^{\circ} = \mathbf{do} \\
= \quad &\mathsf{rand}() \to s \\
&\mathsf{setSeed}(s) \rightsquigarrow () \\
&\mathsf{getSeedA}() \rightsquigarrow s_A \\
&\mathsf{prng}(s_A) \to (s'_A, k_A) \\
&\mathsf{getSeedB}() \rightsquigarrow s_B \\
&\mathsf{prng}(s_B) \to (s'_B, k_B) \\
&\mathsf{return}\ (m, s'_A, m \oplus k_A \oplus k_B, s'_B, m \oplus k_A)
\end{aligned}
$$

STREAM CIPHER IS SECURE



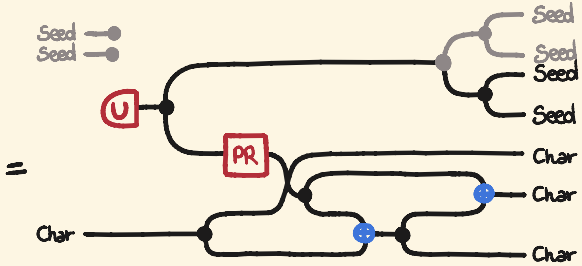


STREAM CIPHER IS SECURE



=

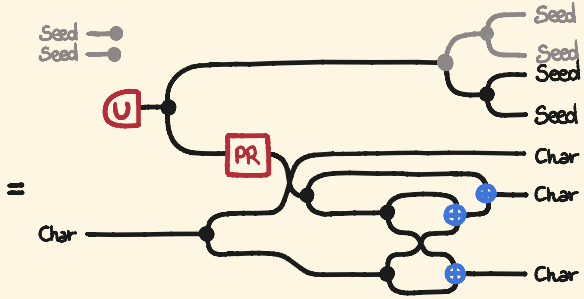
by associativity of copy



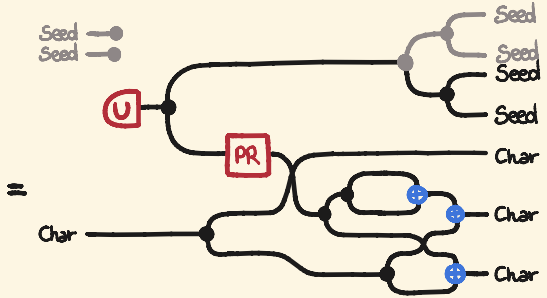
=

pseudorandom is deterministic

STREAM CIPHER IS SECURE

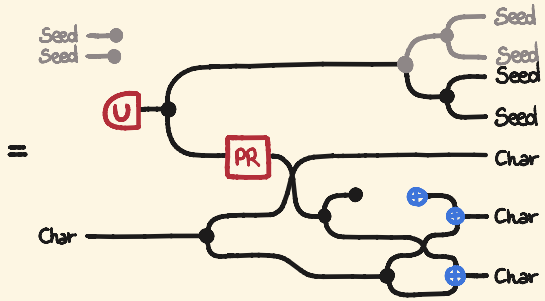


XOR is deterministic

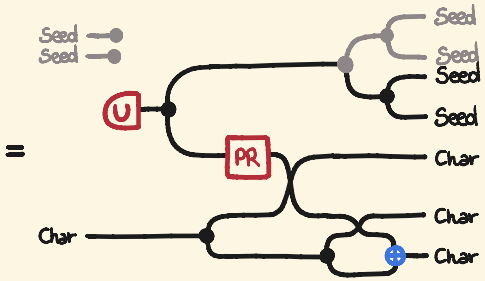


by associativity of copy and XOR

STREAM CIPHER IS SECURE

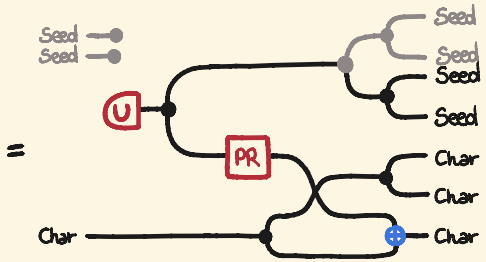


xor is nihilpotent

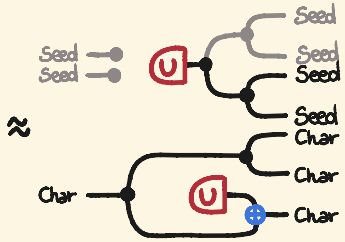


by unitality of copy and xor

STREAM CIPHER IS SECURE

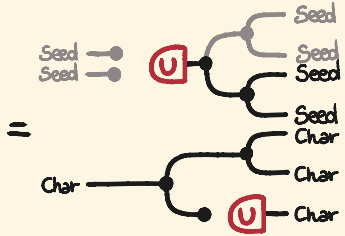


by associativity of copy

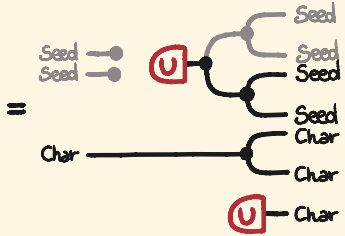


by assumption

STREAM CIPHER IS SECURE



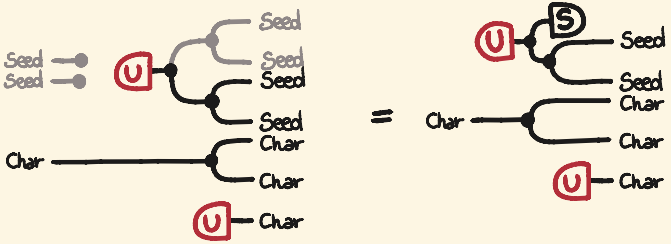
the uniform distribution is a Sweedler integral for xor



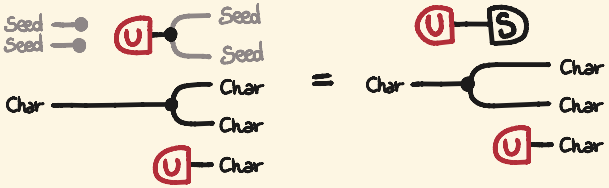
by unitality of copy

STREAM CIPHER IS SECURE

cipher° ≈



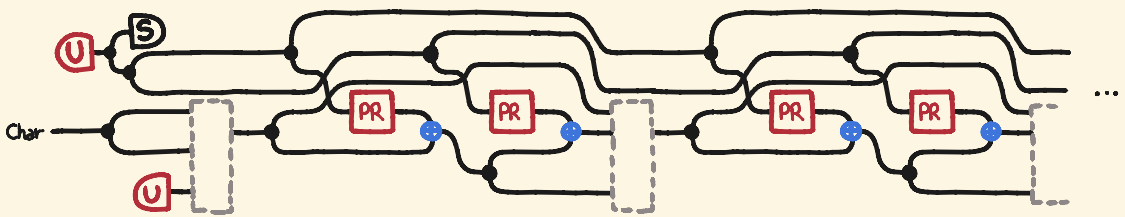
secure° :=



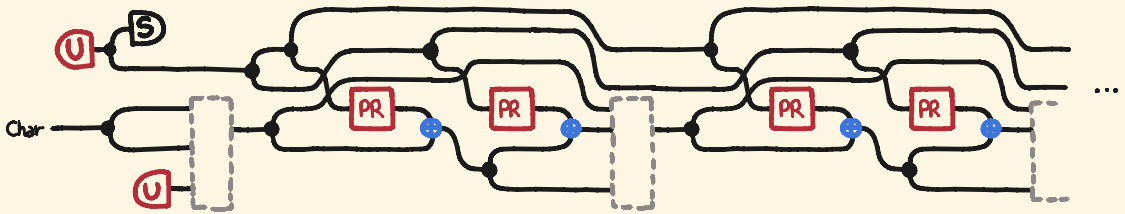
STREAM CIPHER IS SECURE

cipher

=

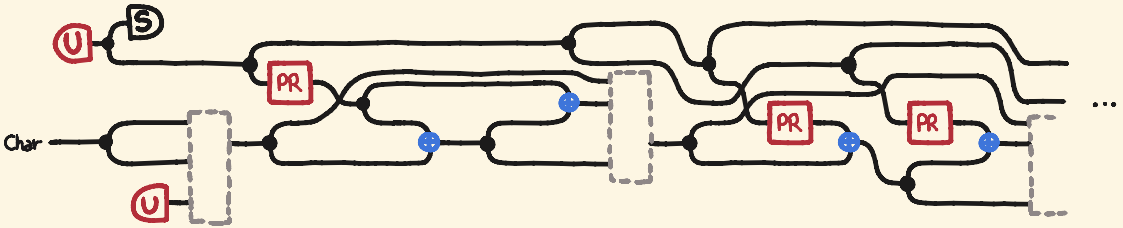


= (by sliding)

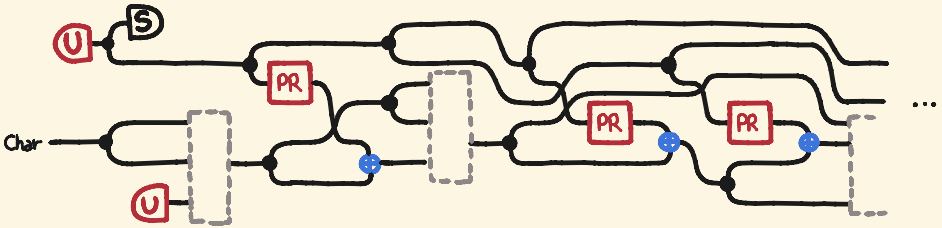


STREAM CIPHER IS SECURE

= (pseudorandom is deterministic)

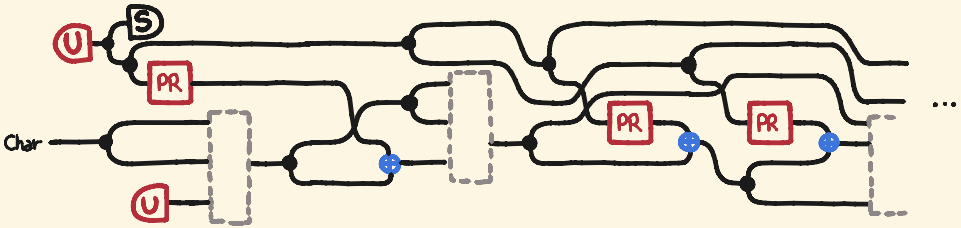


= (XOR is deterministic and nihilpotent)

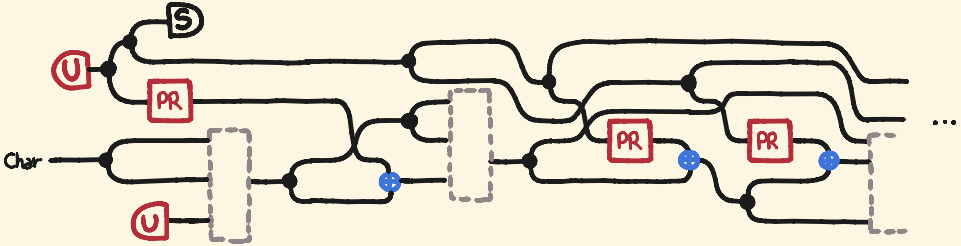


STREAM CIPHER IS SECURE

= (by sliding)

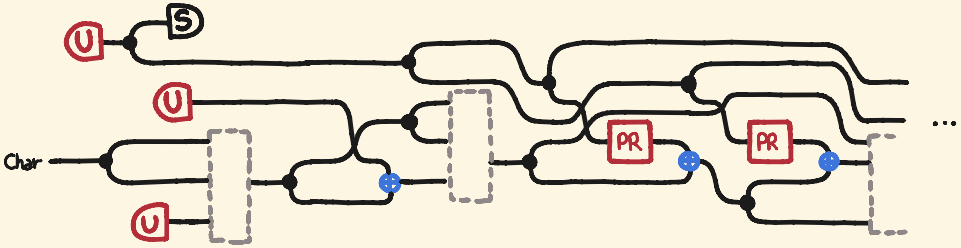


= (by associativity)

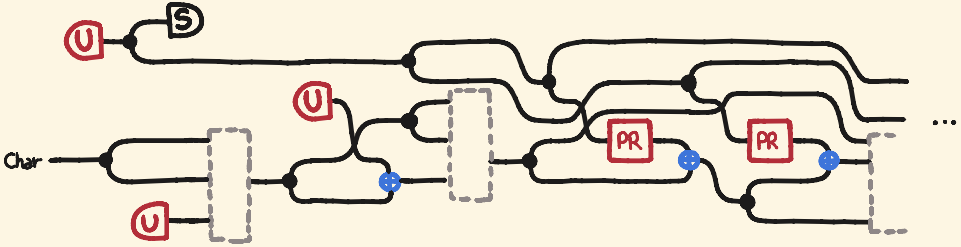


STREAM CIPHER IS SECURE

≈ (by assumption on pseudorandom)

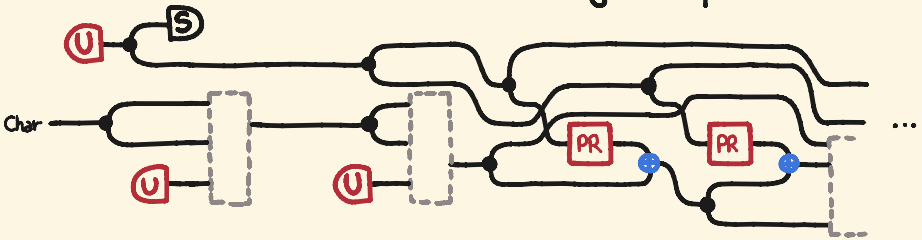


= (by sliding)

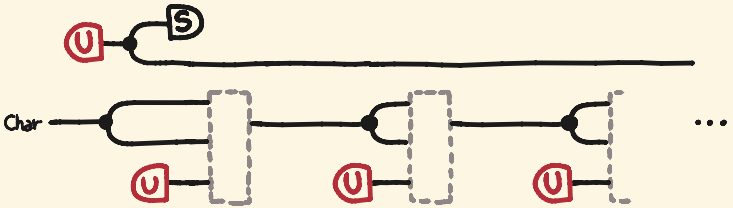


STREAM CIPHER IS SECURE

= (unif is a Sweedler integral for xor)

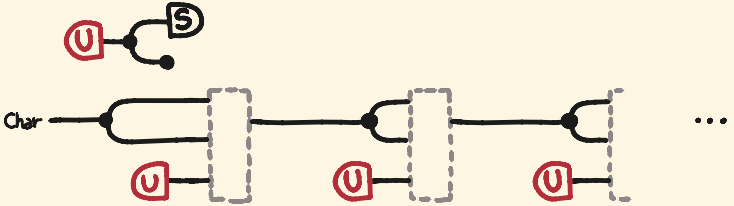


≈ (by coinduction)

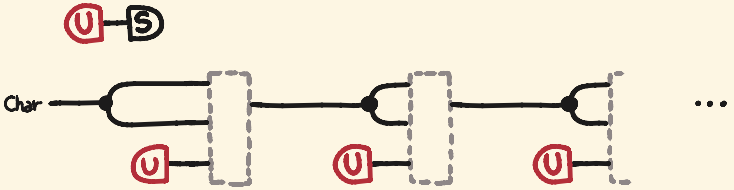


STREAM CIPHER IS SECURE

= (by coinduction)



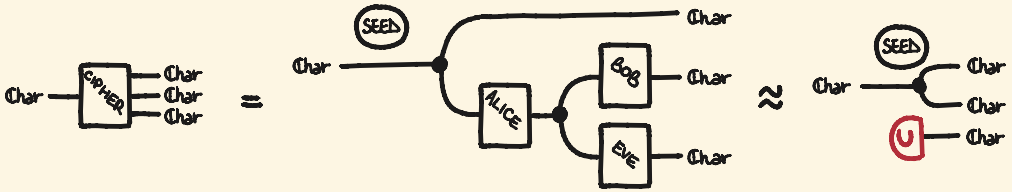
= (by unitality)



= secure

STREAM CIPHER IS SECURE

We have shown



using sliding and coinduction.