A General Account of Coinduction Up-To

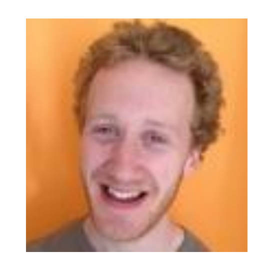
Filippo Bonchi
CNRS, Ens-Lyon

Joint work with Daniela Petrisan, Damien Pous and Jurriaan Rot

A Fruitful Approach

A Fruitful Approach

10-15/07/2016
Shangai



Coinduction (lattice theoretic)

<u>Knaster-Tarski fixed point:</u>

L a complete lattice and B:**L**-->**L** a monotone map

$\nu B = \bigcup \{x \mid x \subseteq B(x)\}$

Coinduction (lattice theoretic)

<u>Knaster-Tarski fixed point:</u>

L a complete lattice and B:**L**-->**L** a monotone map

$\nu B = \bigcup \{x \mid x \subseteq B(x)\}$

$$\frac{\exists y,\ \ x \subseteq y \subseteq B(y)}{x \subseteq \nu B}$$

Coinduction (lattice theoretic)

<u>Knaster-Tarski fixed point:</u>

L a complete lattice and B:**L**-->**L** a monotone map

$\nu B = \bigcup\{x \mid x \subseteq B(x)\}$

$$\frac{\exists y,\ \ x \subseteq y \subseteq B(y)}{x \subseteq \nu B}$$

The post fixed points of B are called *invariants* or *bisimulations*

Language Equivalence

A Deterministic Automaton (DA) is a triple (X,o,t)

- X is the set of states
- o:X-->2 is the output function
- t:X-->X^A is the transition function

(a coalgebra for the functor 2xId^A)

Language Equivalence

A Deterministic Automaton (DA) is a triple (X,o,t)

- X is the set of states
- o:X-->2 is the output function
- t:X-->X^A is the transition function

(a coalgebra for the functor $2 \times Id^A$)

B:$\mathbf{Rel_X}$-->$\mathbf{Rel_X}$

B(R)={(x,y) | o(x)=o(y) and for all a∈A t(x)(a) R t(y)(a)}

Language Equivalence

A Deterministic Automaton (DA) is a triple (X,o,t)

- X is the set of states
- o:X-->2 is the output function
- t:X-->X^A is the transition function

(a coalgebra for the functor 2xId^A)

B:$\mathbf{Rel_X}$-->$\mathbf{Rel_X}$

B(R)={(x,y) | o(x)=o(y) and for all a∈A t(x)(a) R t(y)(a)}

Language equivalence (∼) is νB

Language Equivalence

A Deterministic Automaton (DA) is a triple (X,o,t)

- X is the set of states
- o:X-->2 is the output function
- t:X-->X^A is the transition function

(a coalgebra for the functor $2 \times Id^A$)

B:$\mathbf{Rel_X}$-->$\mathbf{Rel_X}$

B(R)={(x,y) | o(x)=o(y) and for all a∈A t(x)(a) R t(y)(a)}

Language equivalence (∼) is νB

By coinduction, to show x∼y
is enough to find R such that
{(x,y)} ⊆ R and
R ⊆B(R)

Language Equivalence

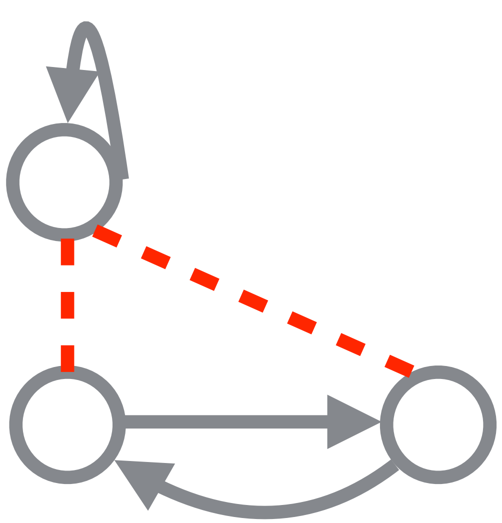
A Deterministic Automaton (DA) is a triple (X,o,t)

- X is the set of states
- o:X-->2 is the output function
- t:X-->X^A is the transition function

(a coalgebra for the functor $2 \times Id^A$)

B:$\mathbf{Rel_X}$-->$\mathbf{Rel_X}$

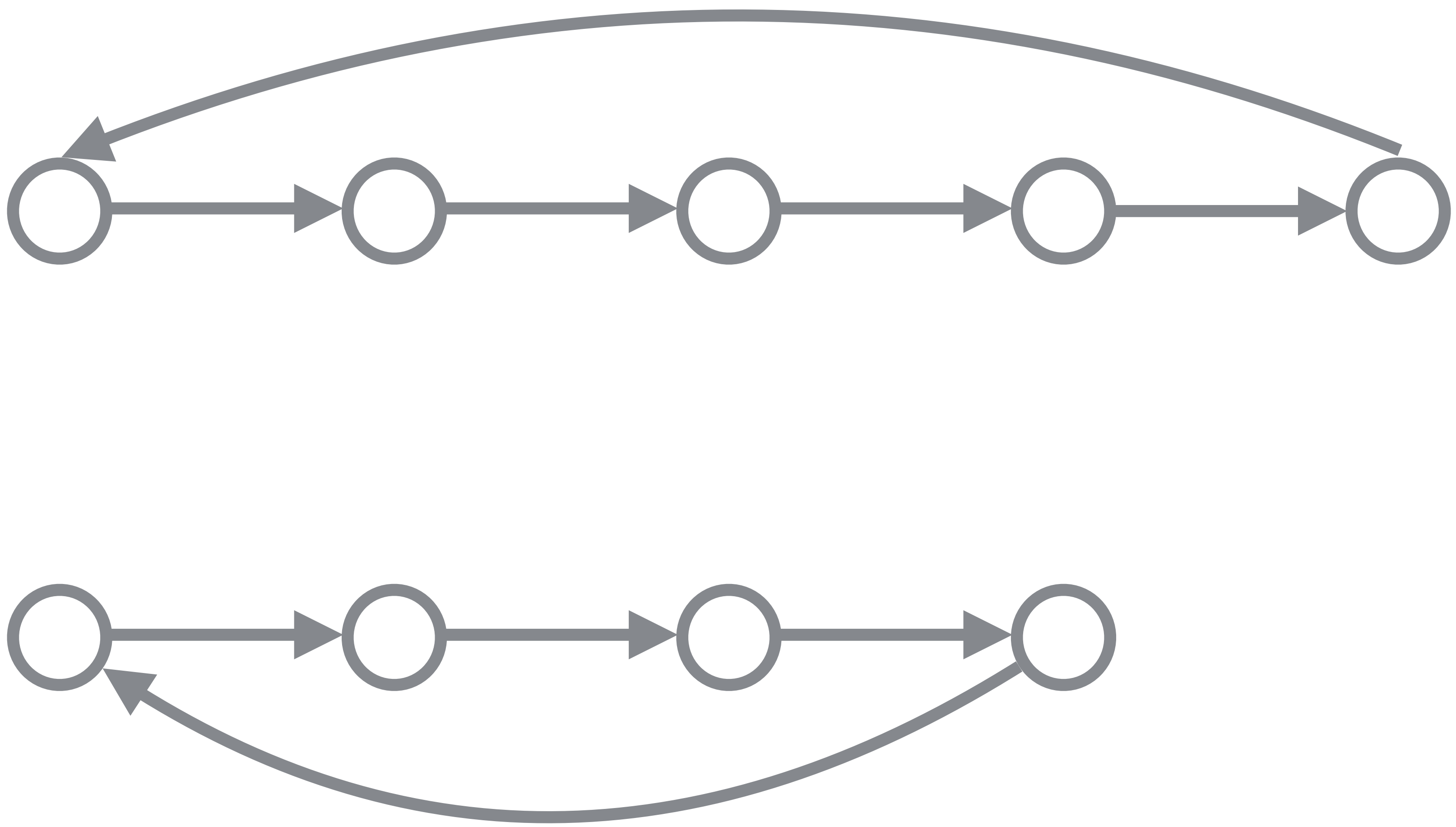
B(R)={(x,y) | o(x)=o(y) and for all a∈A t(x)(a) R t(y)(a)}



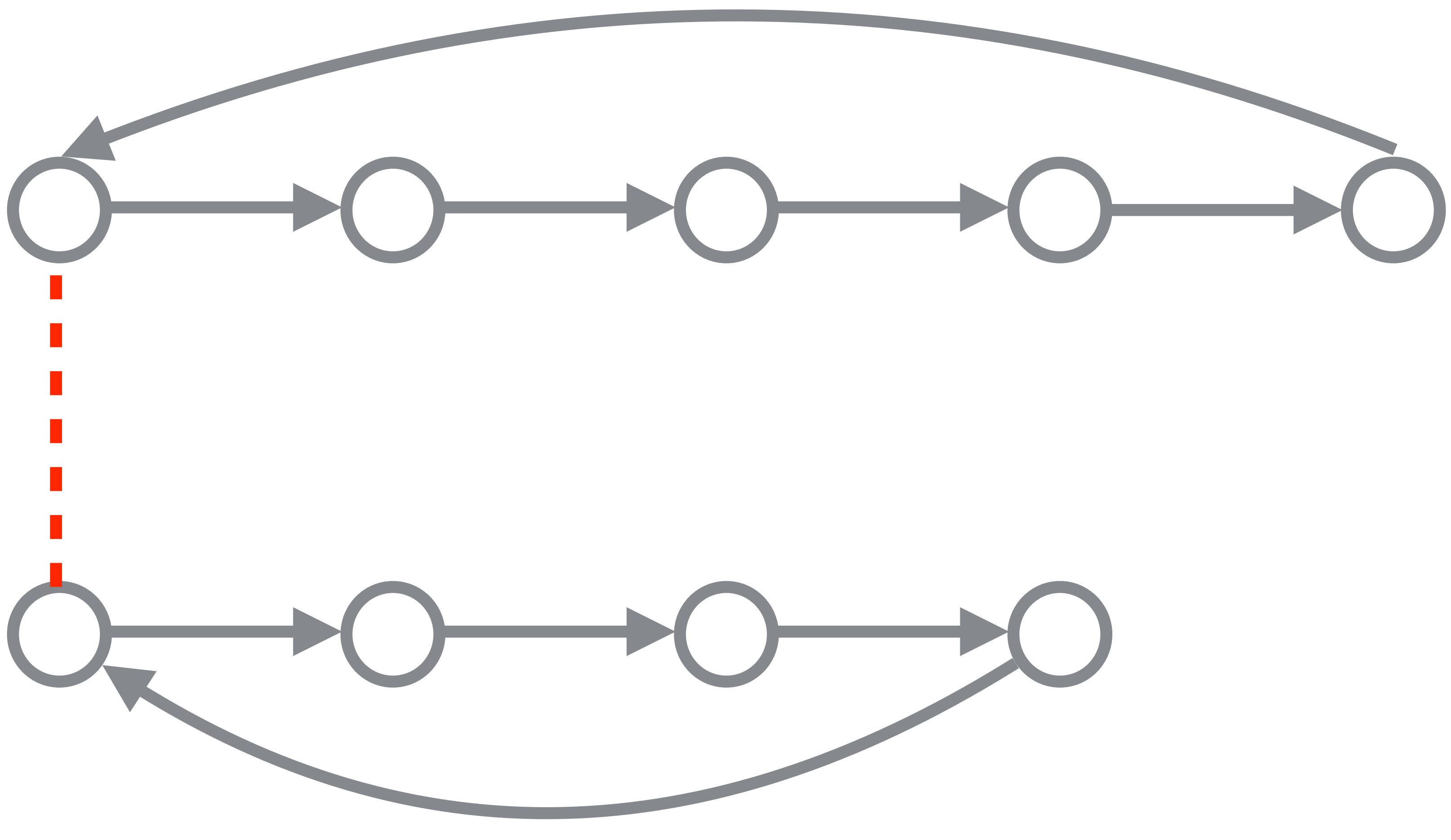
Language equivalence (∼) is νB

By coinduction, to show x∼y
is enough to find R such that
{(x,y)} ⊆ R and
R ⊆B(R)

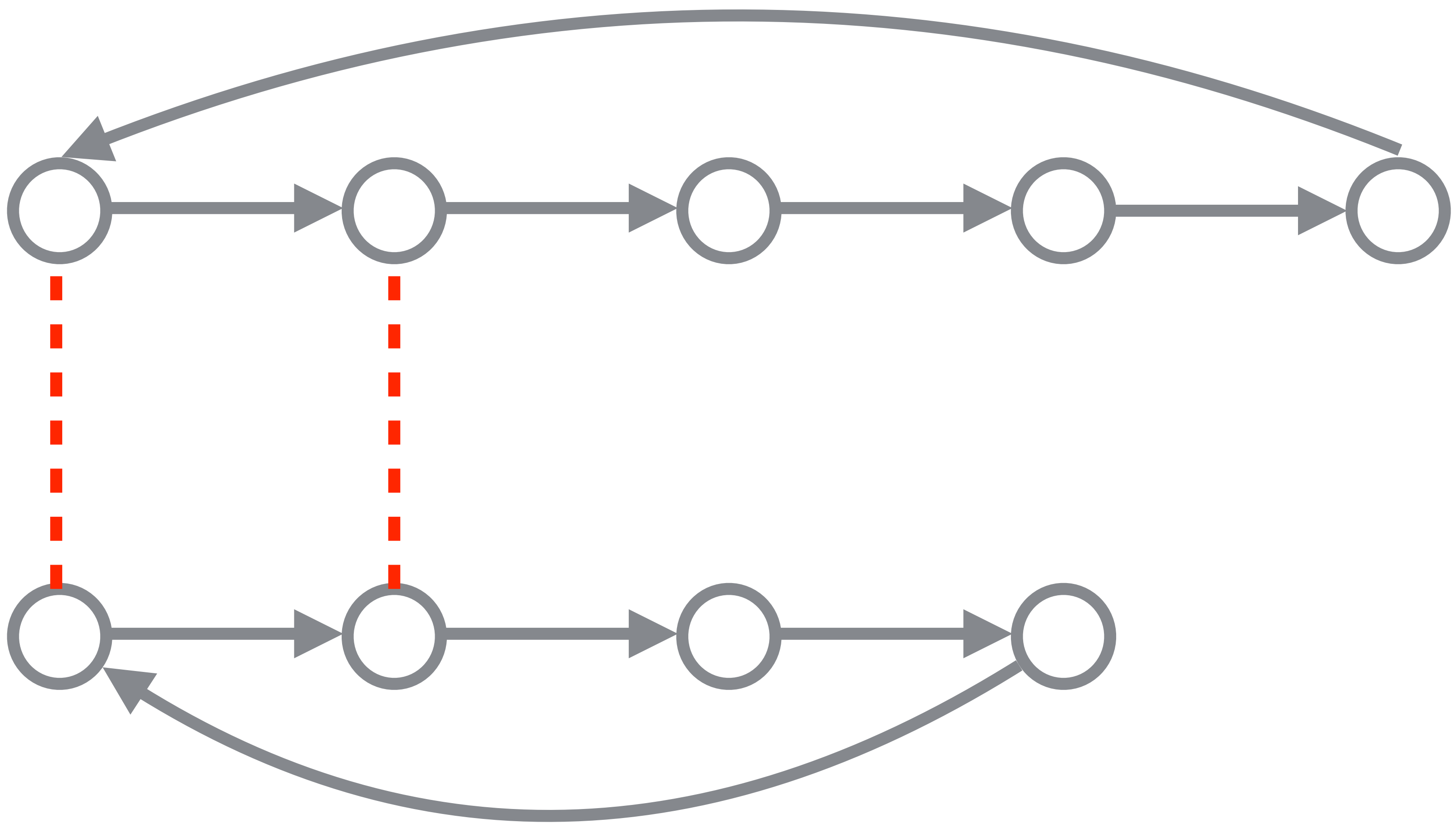
Naive Algorithm



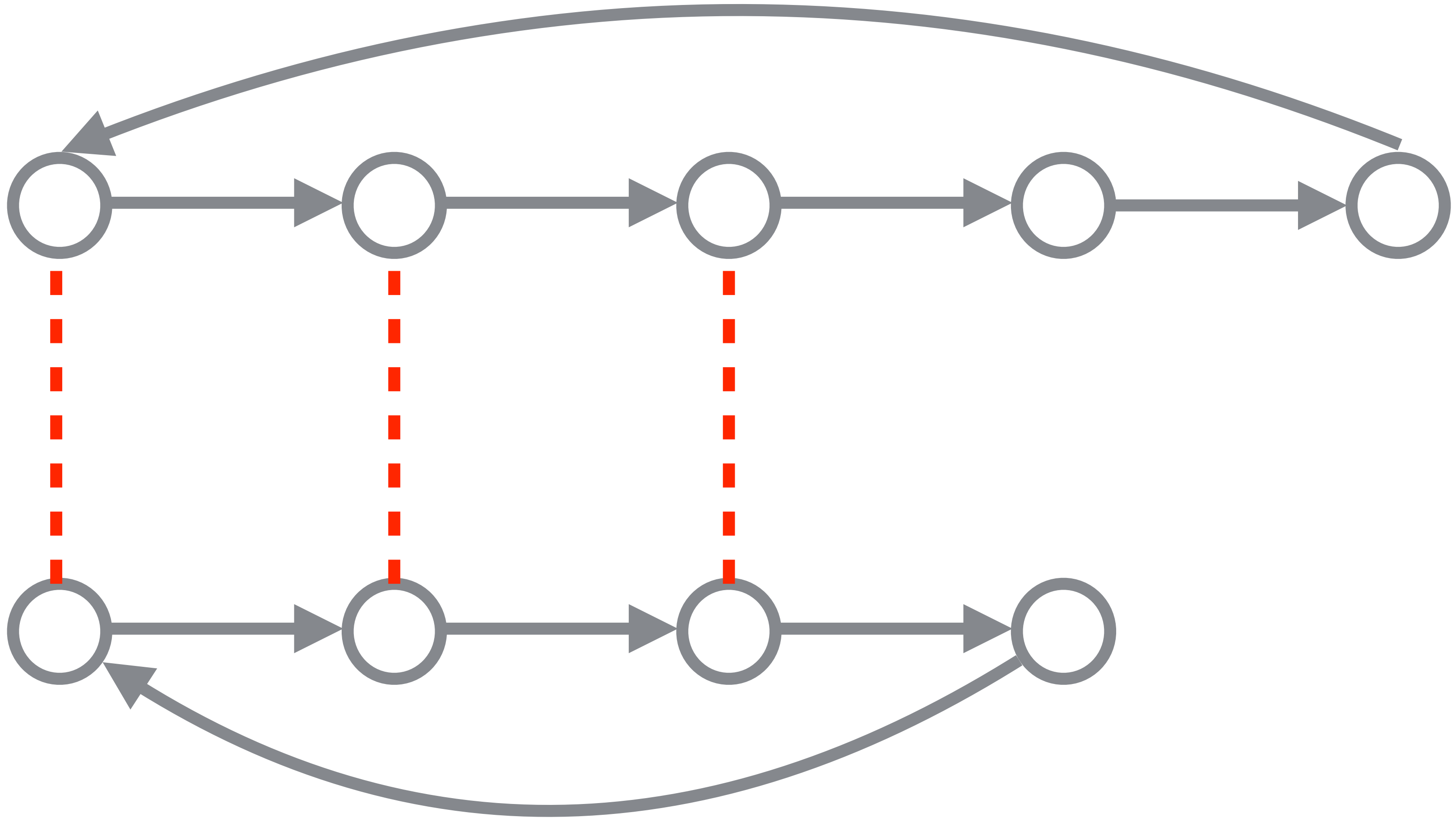
Naive Algorithm



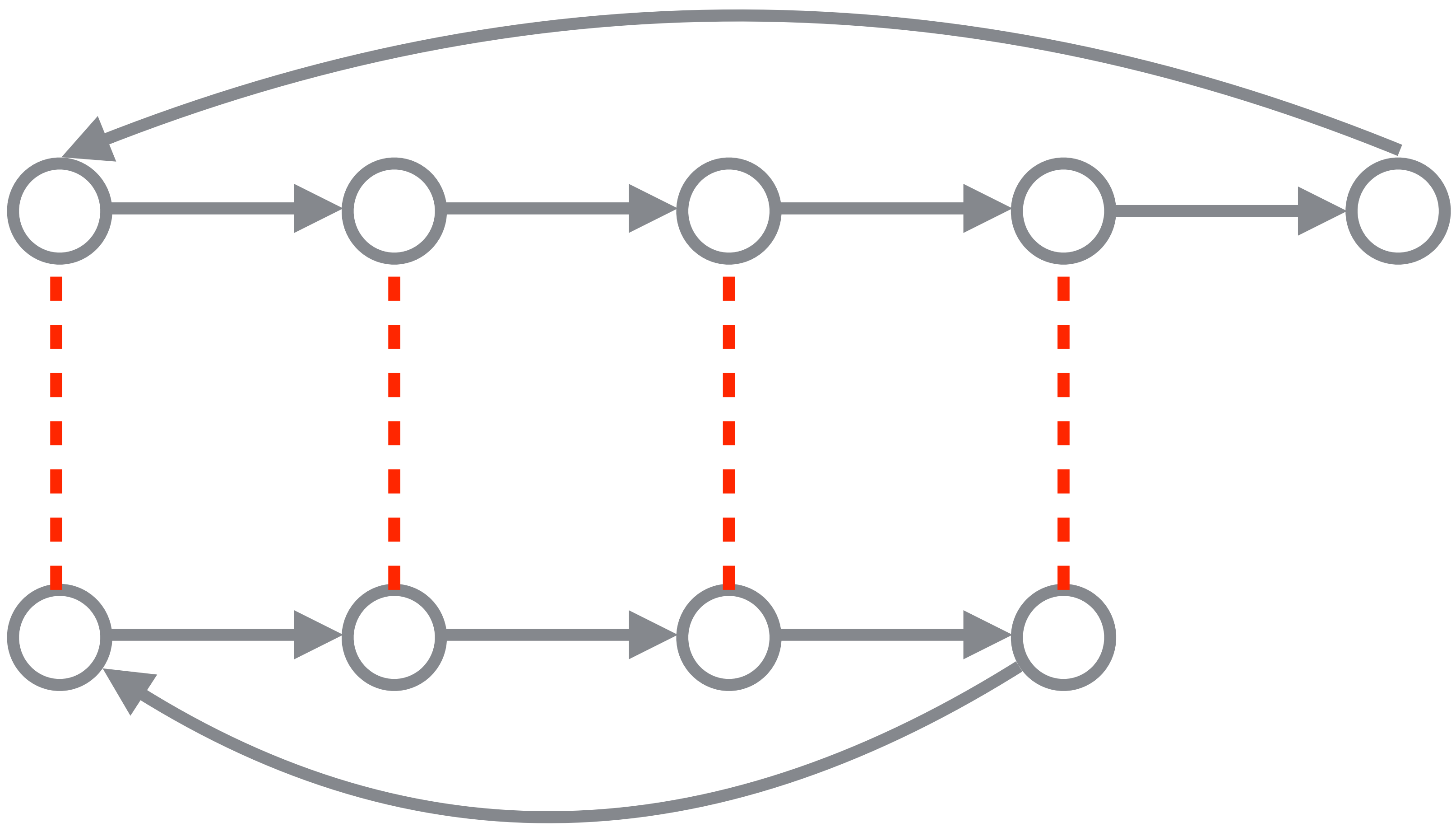
Naive Algorithm



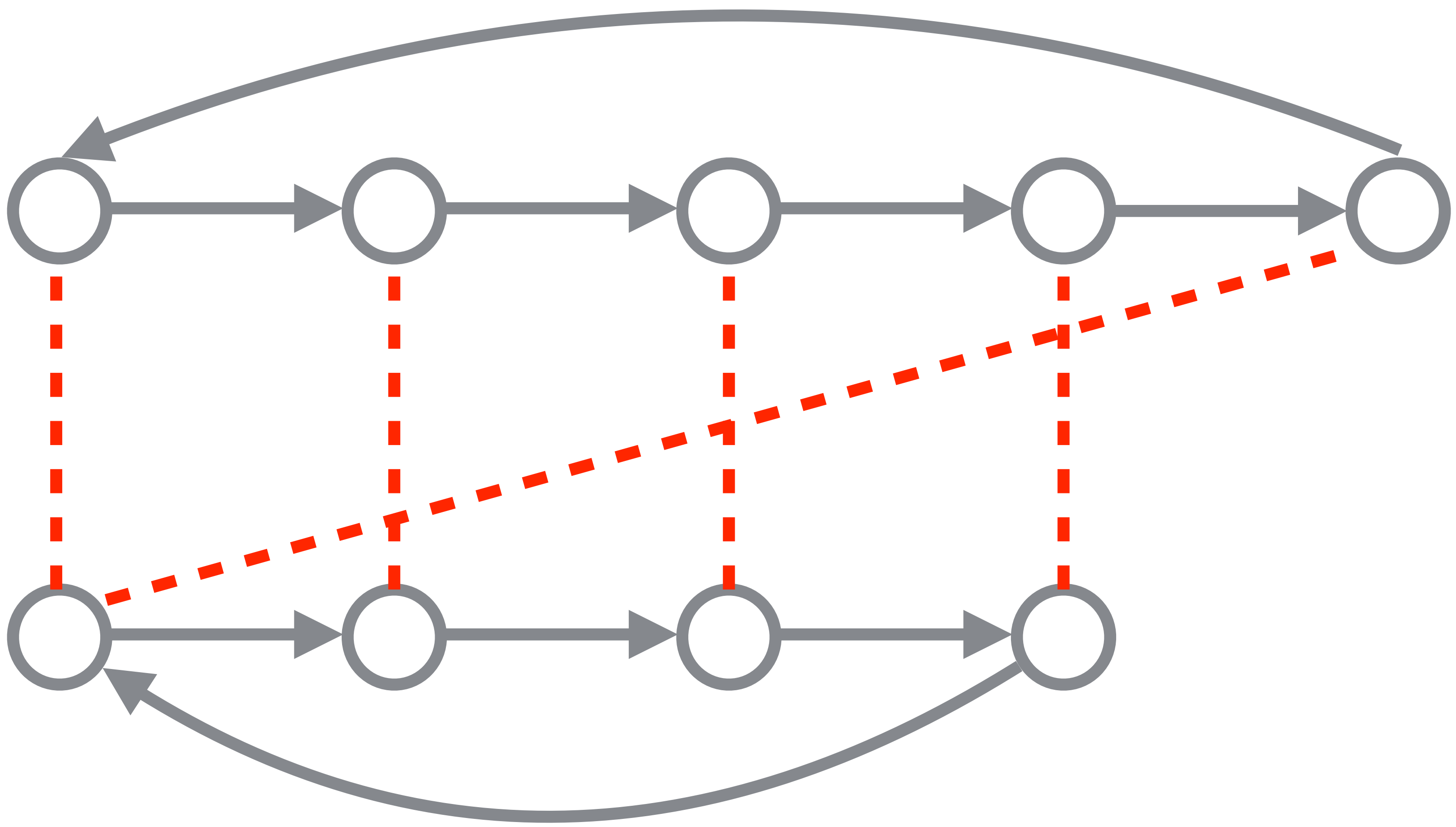
Naive Algorithm



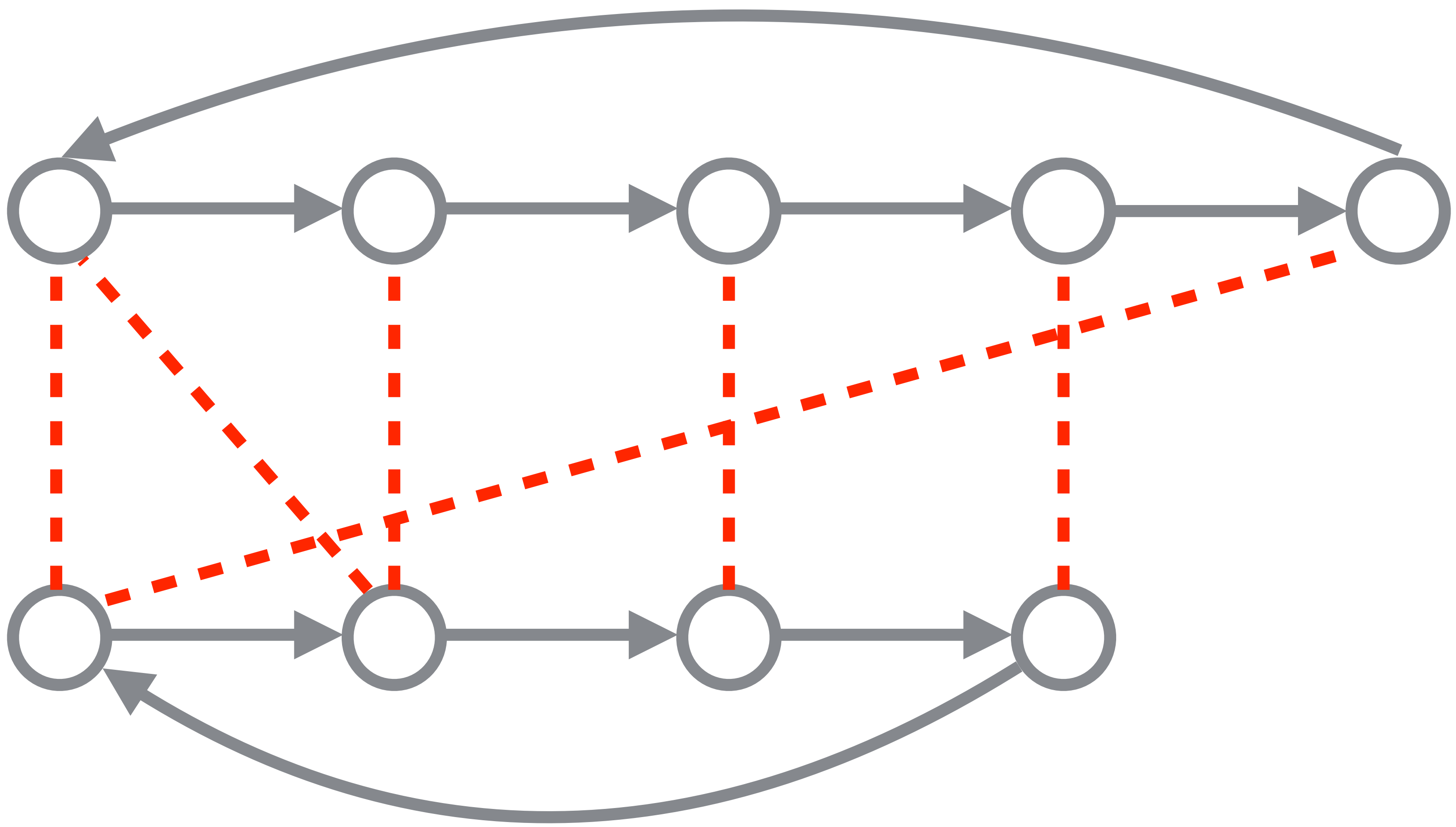
Naive Algorithm



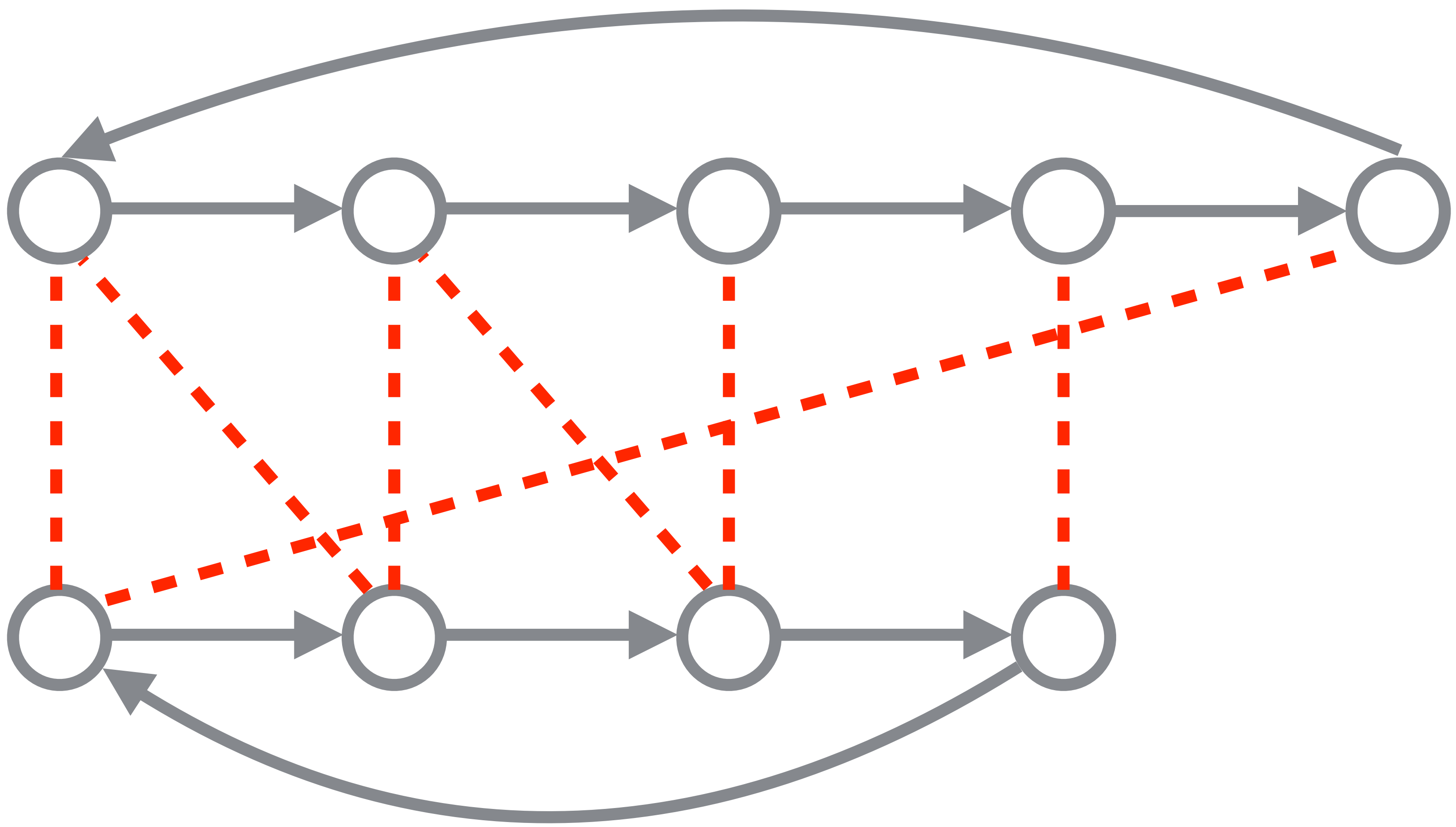
Naive Algorithm



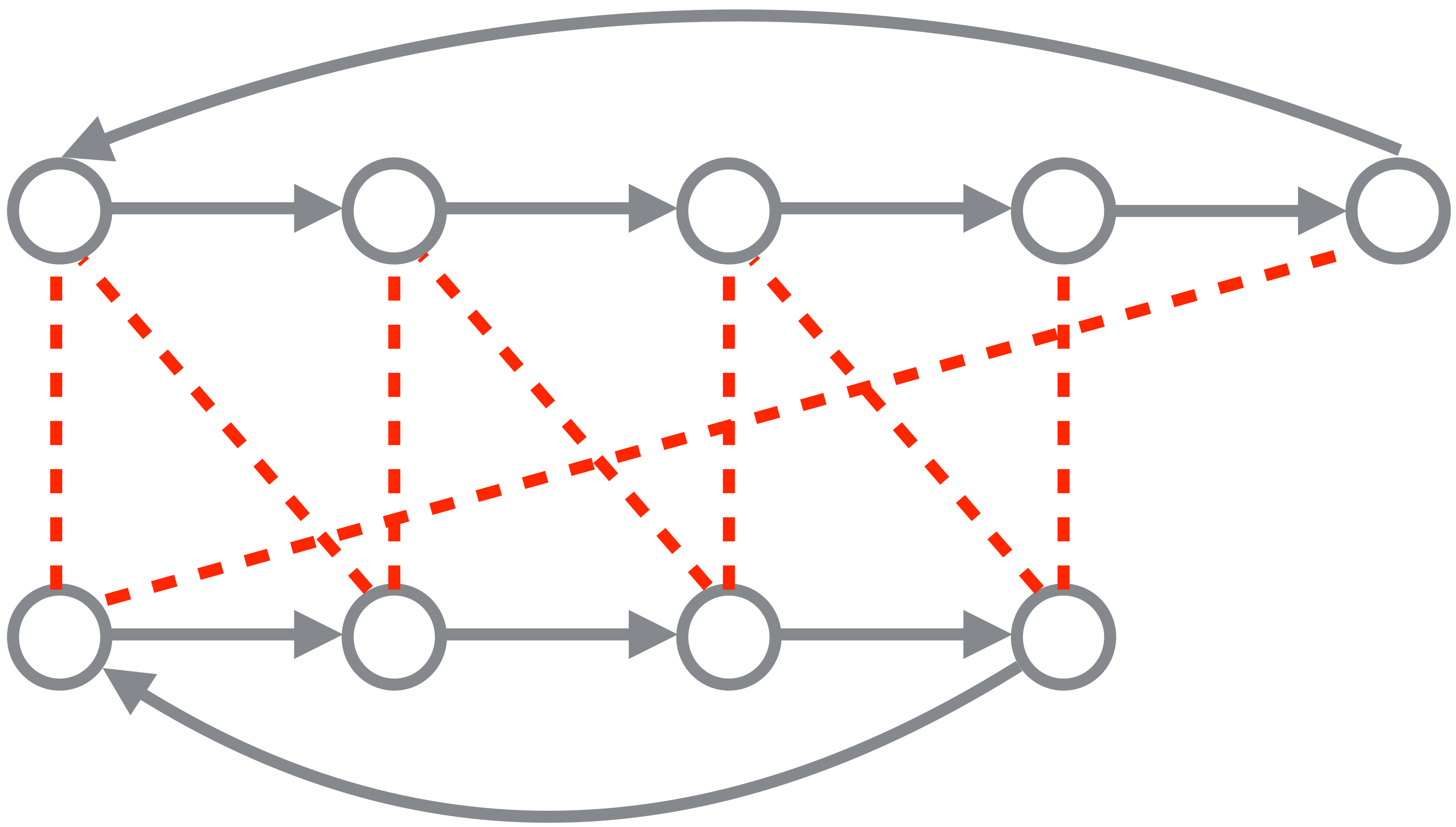
Naive Algorithm



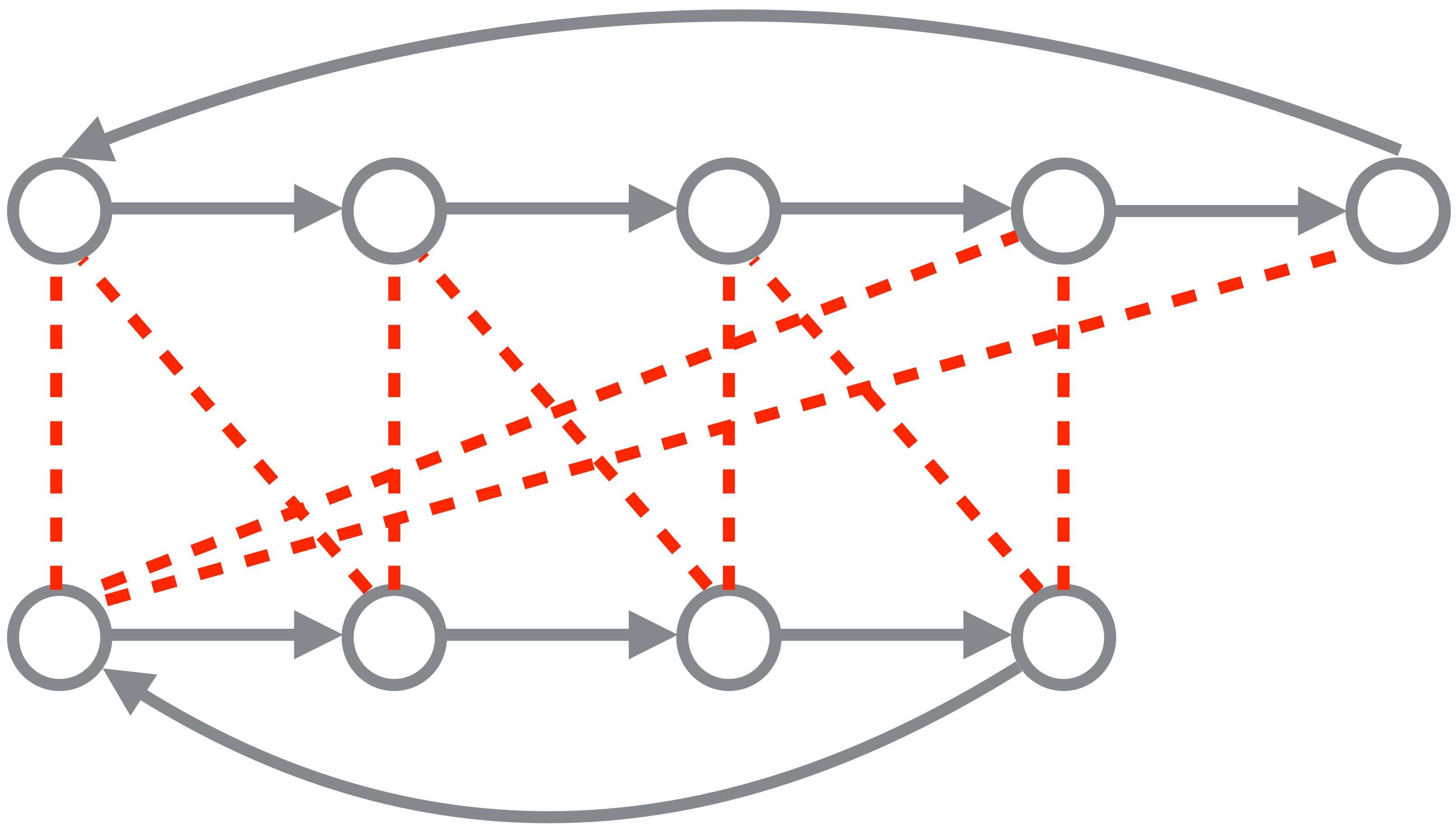
Naive Algorithm



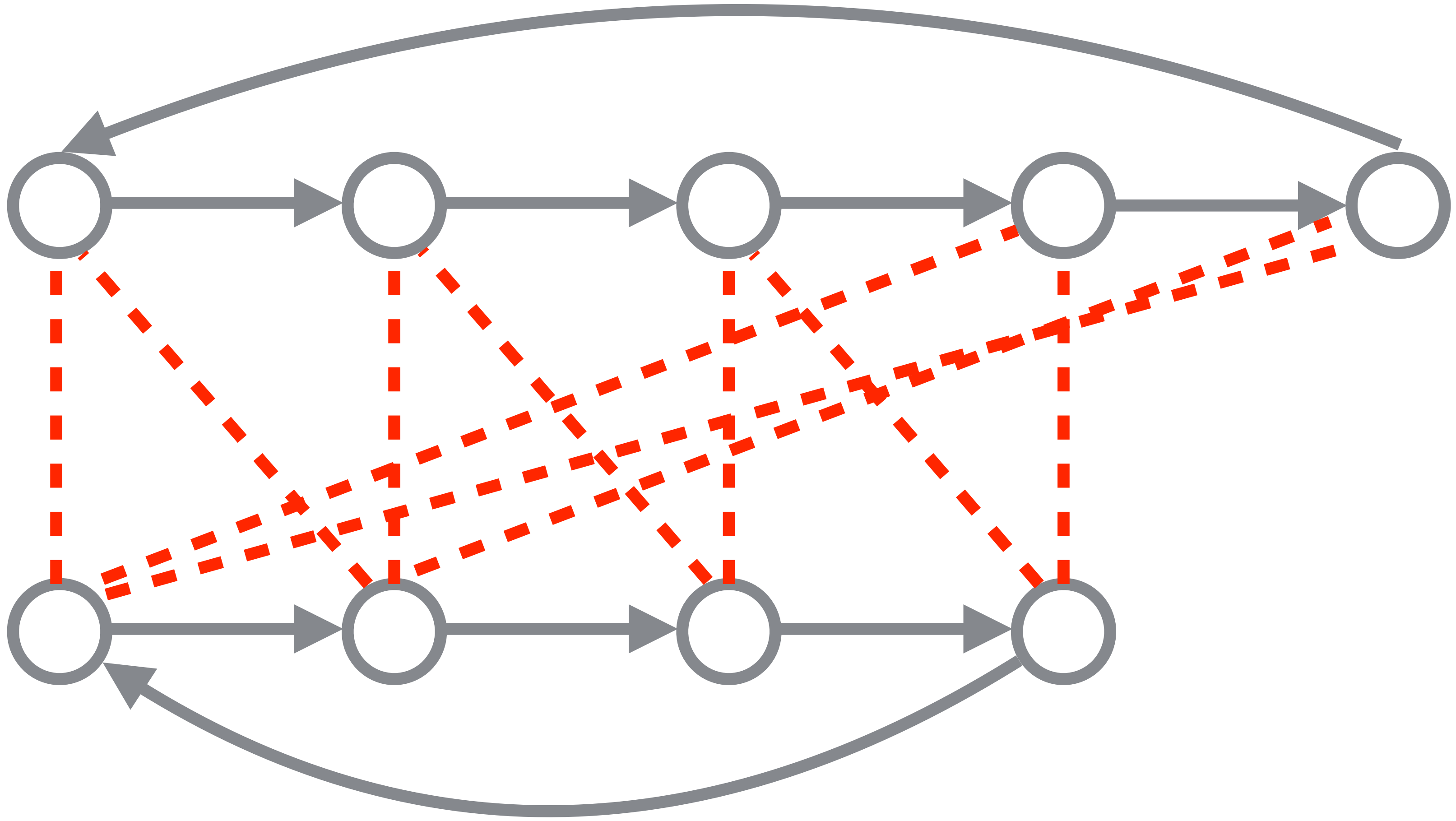
Naive Algorithm



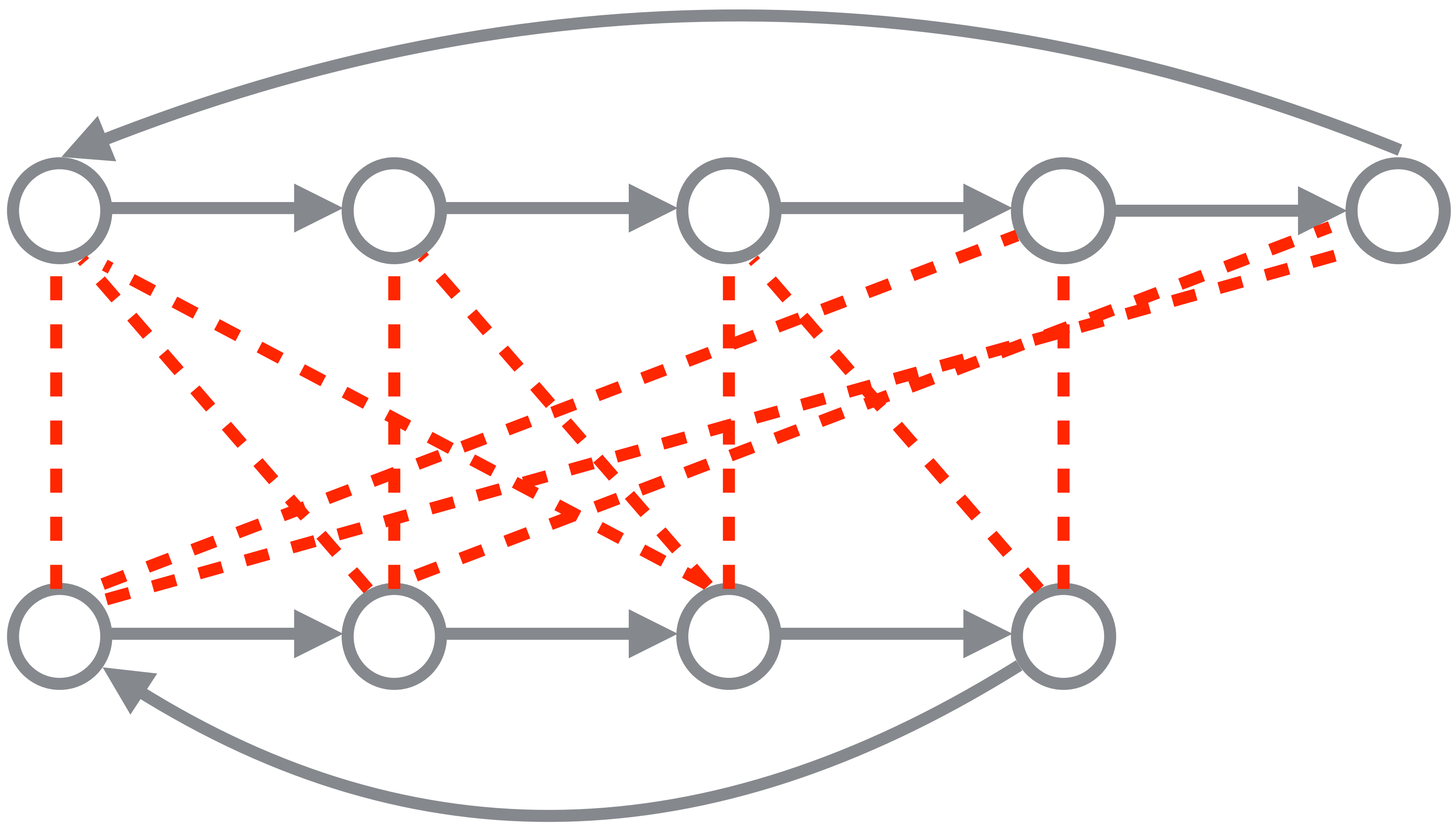
Naive Algorithm



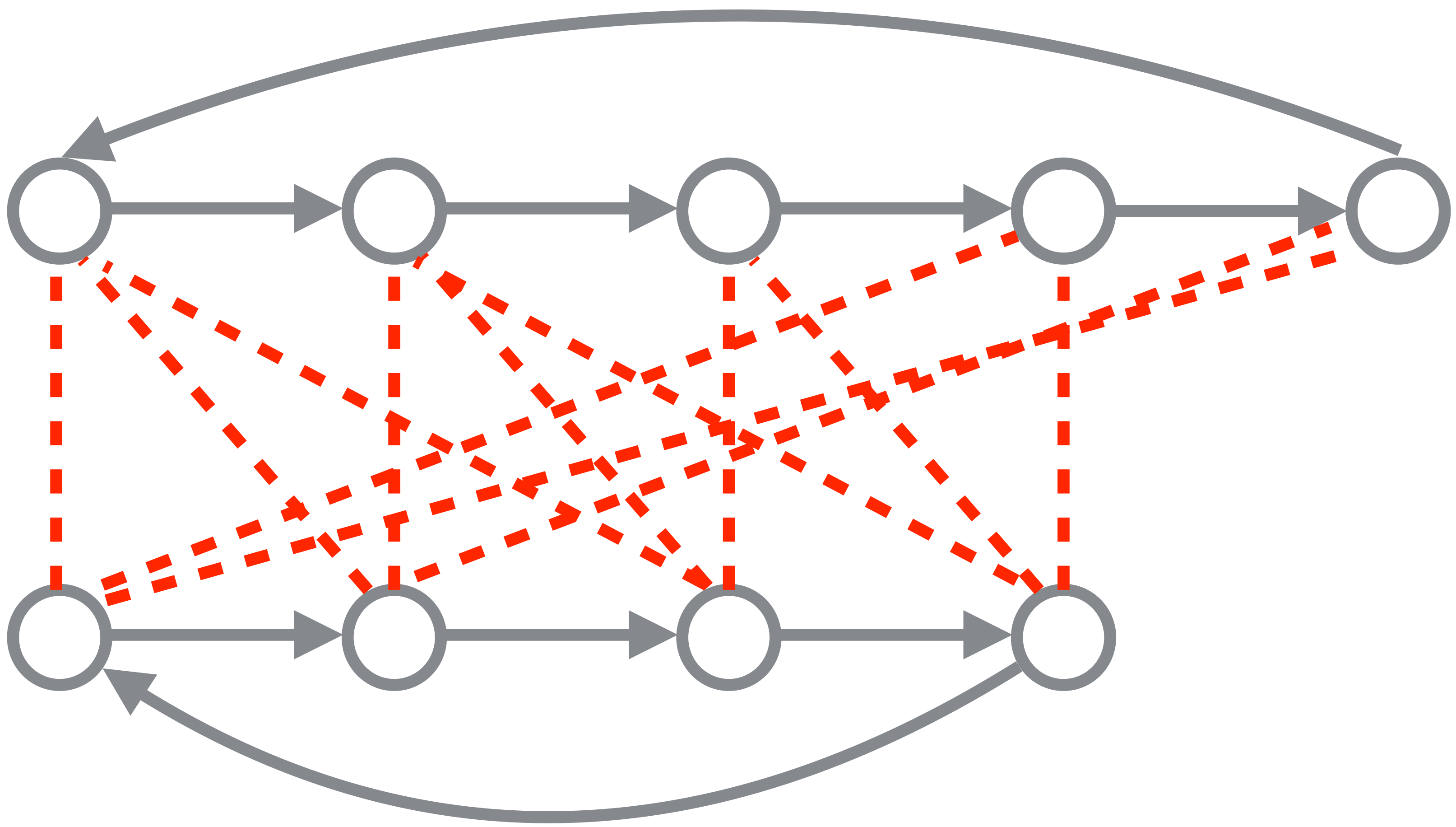
Naive Algorithm



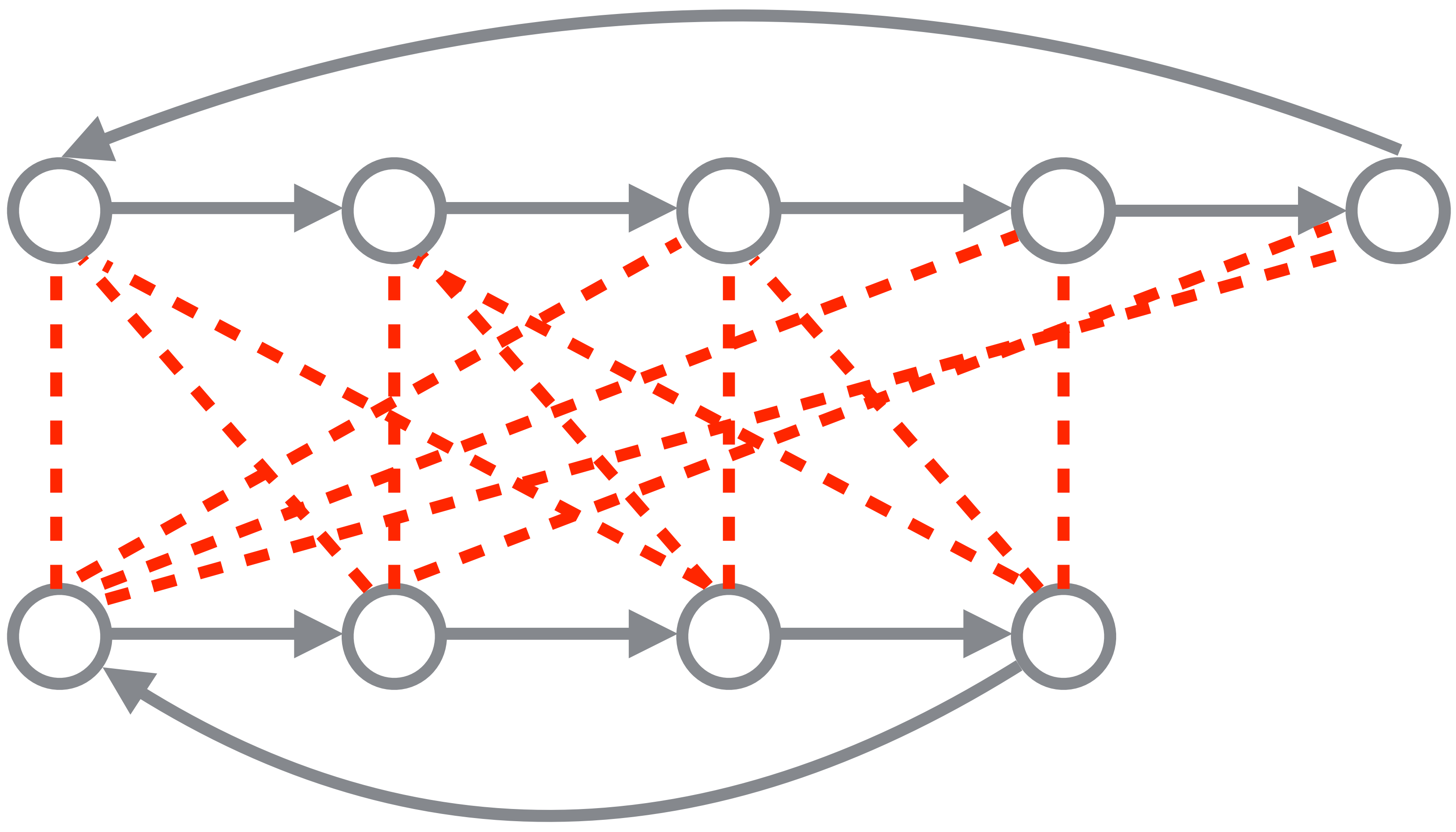
Naive Algorithm



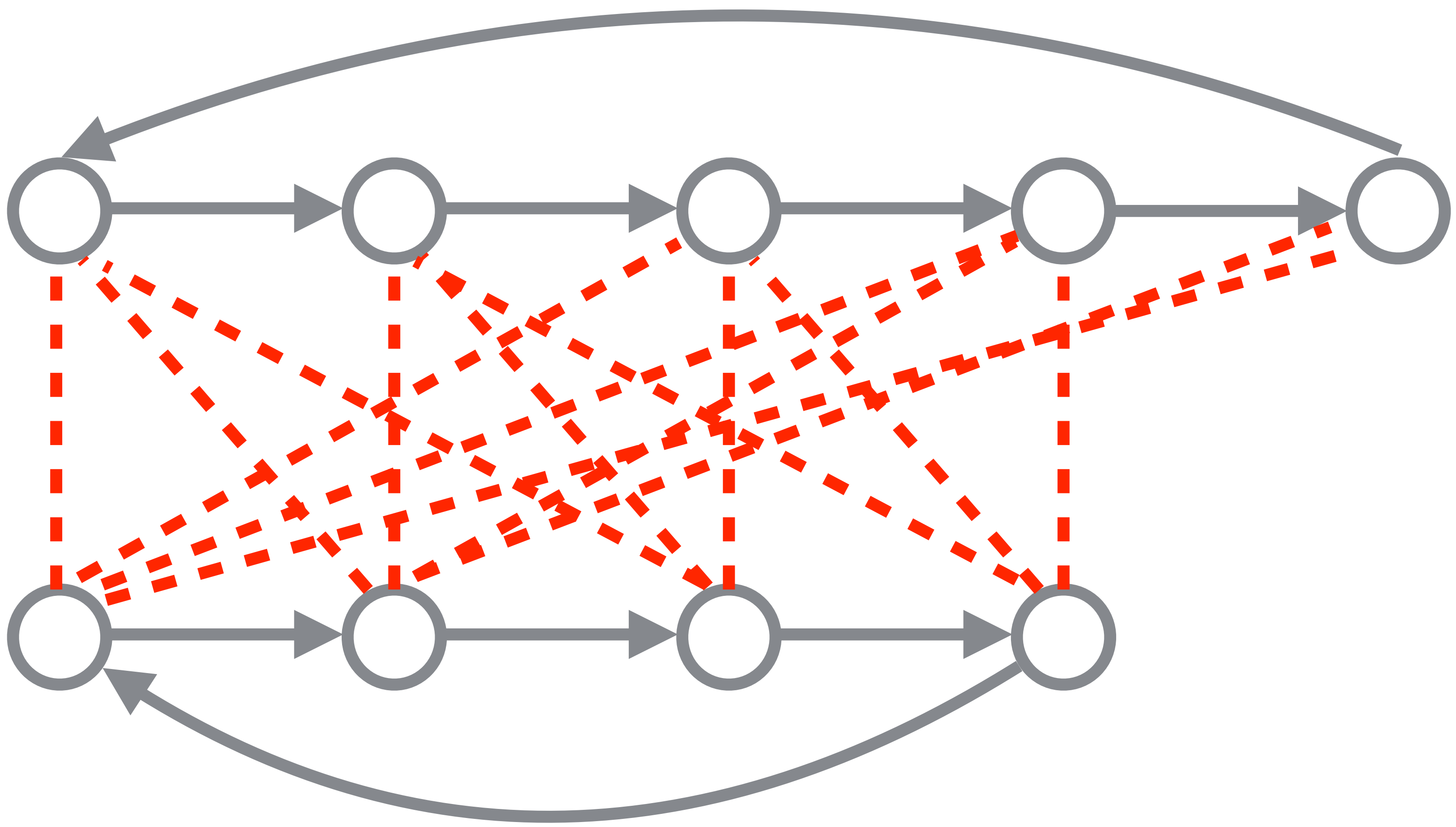
Naive Algorithm



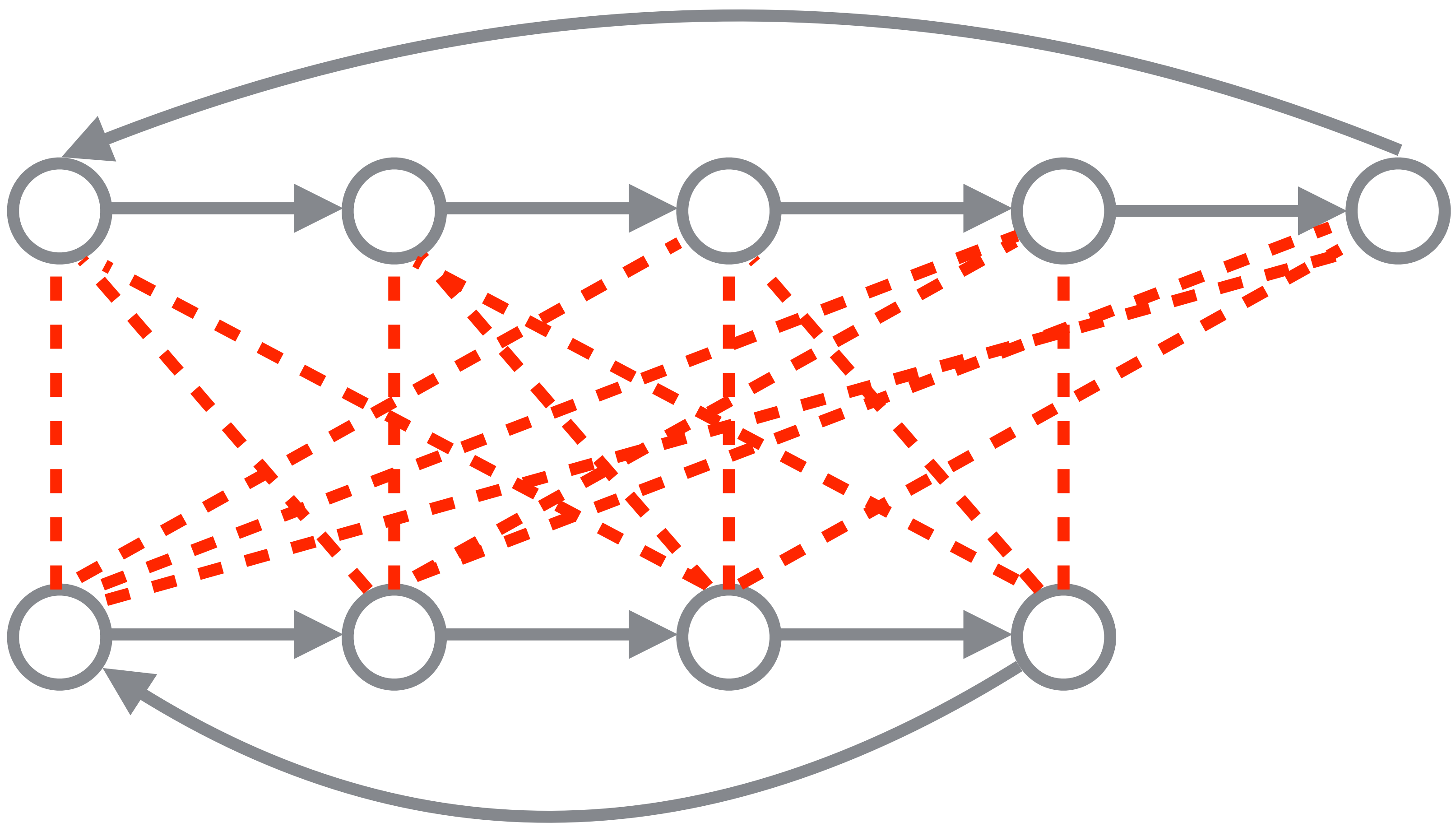
Naive Algorithm



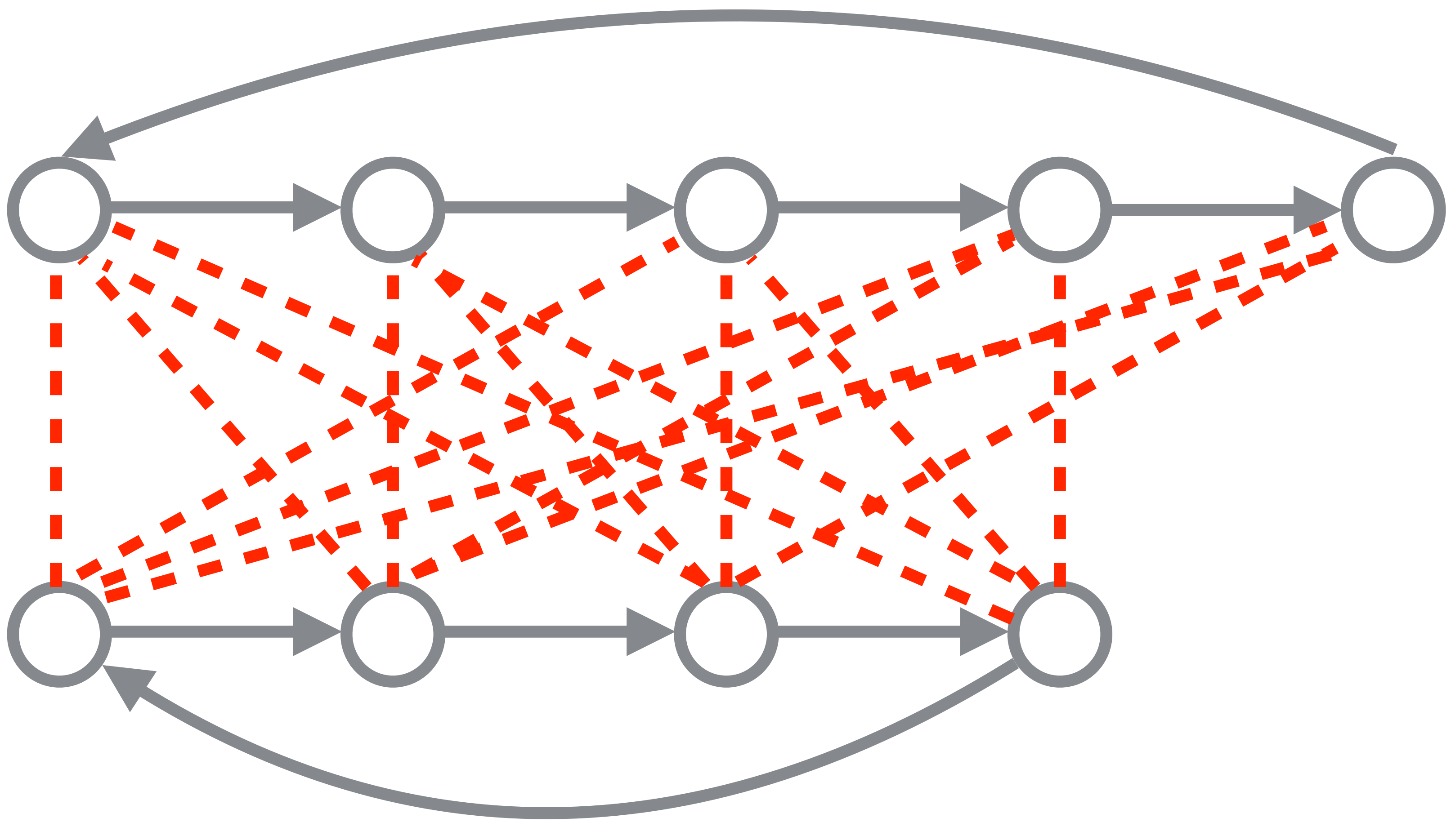
Naive Algorithm



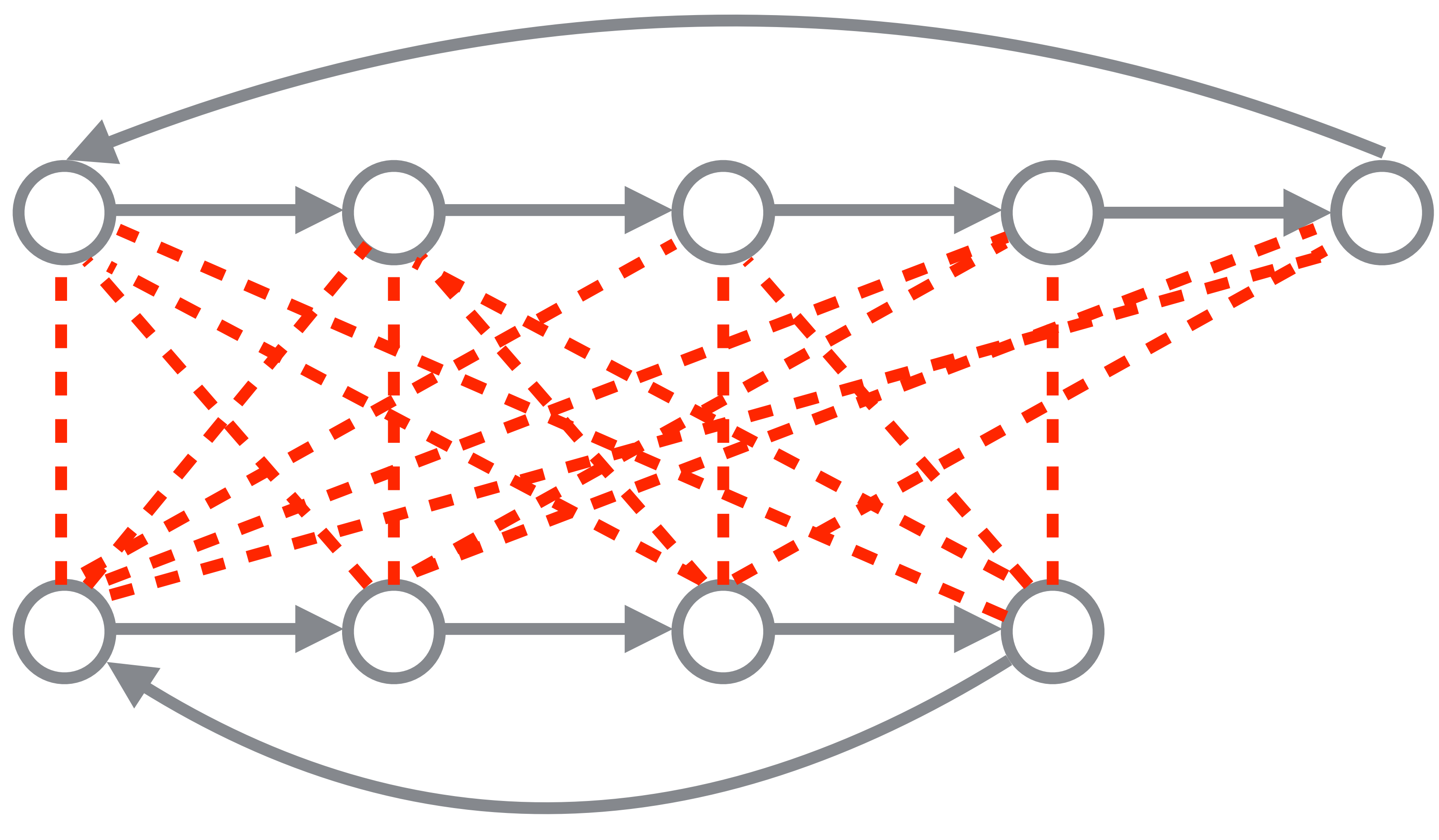
Naive Algorithm



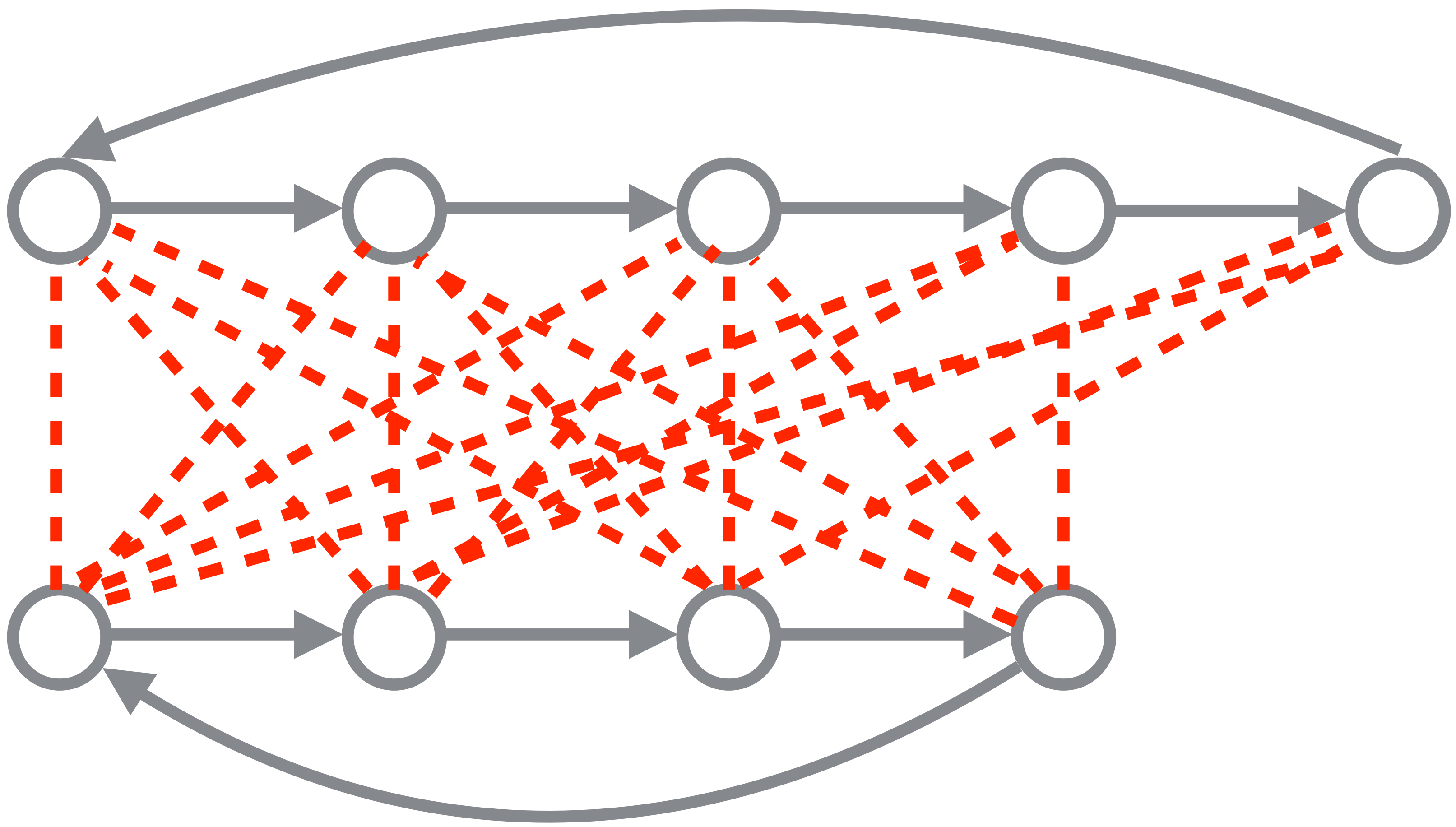
Naive Algorithm



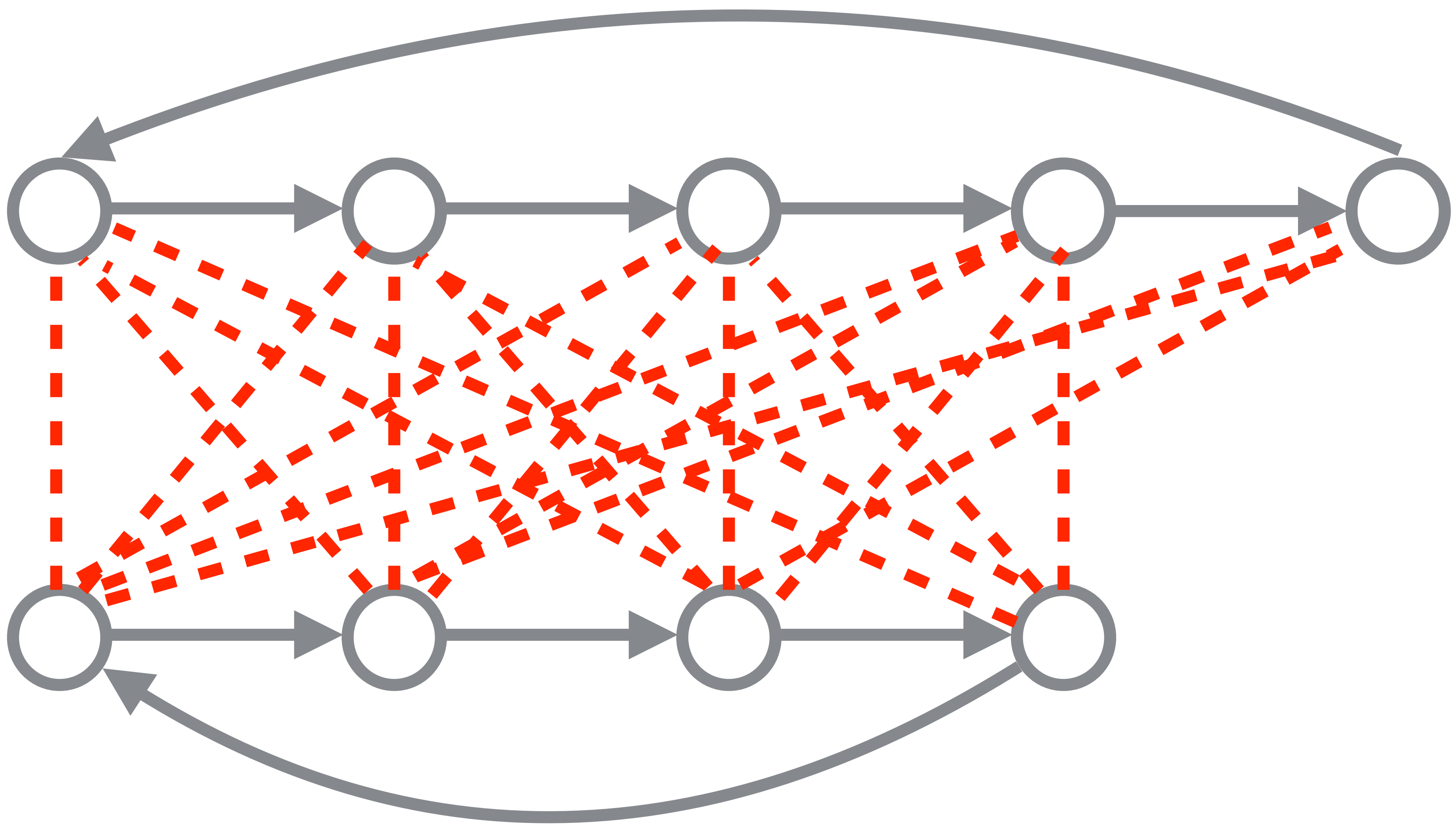
Naive Algorithm



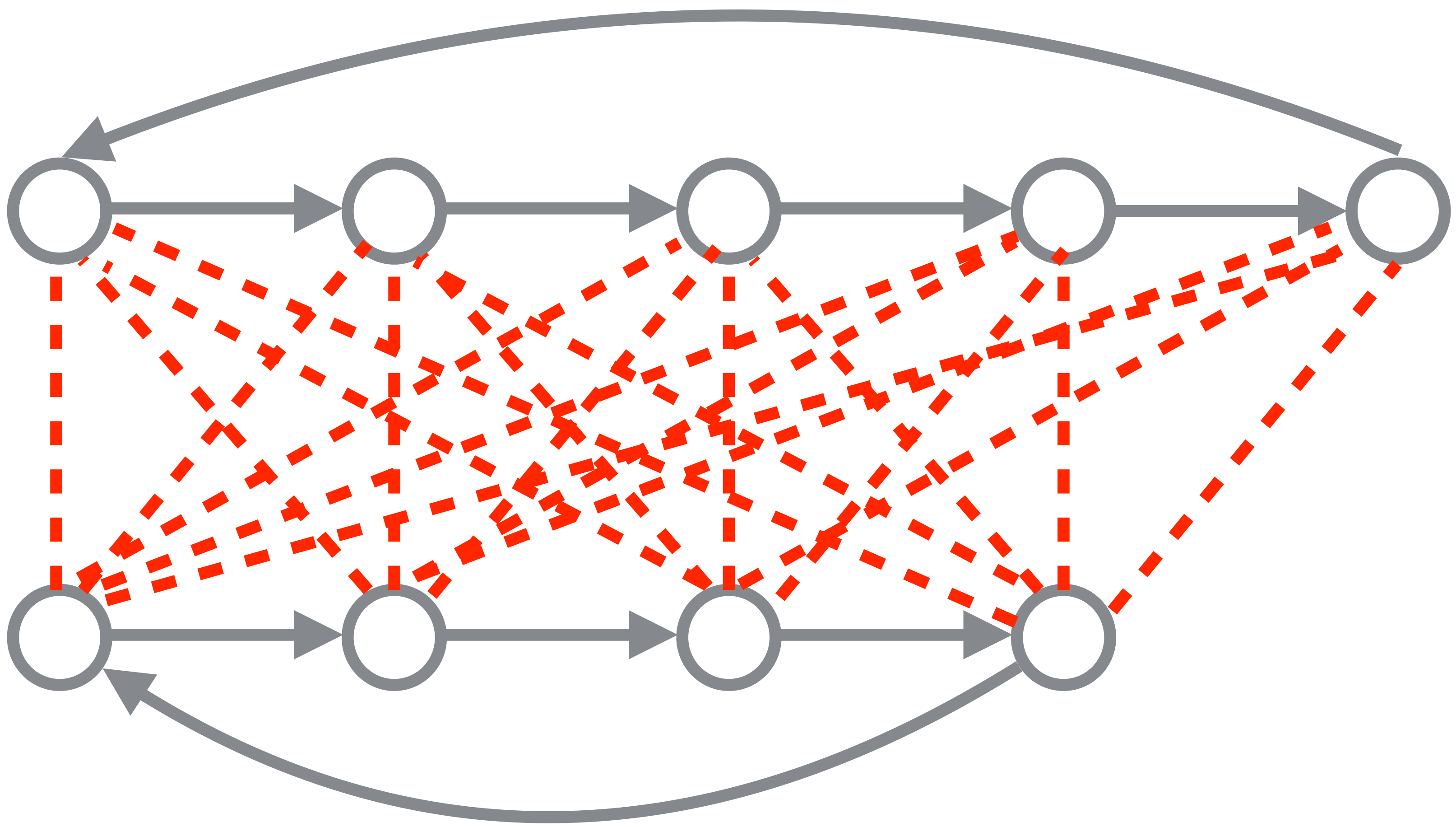
Naive Algorithm



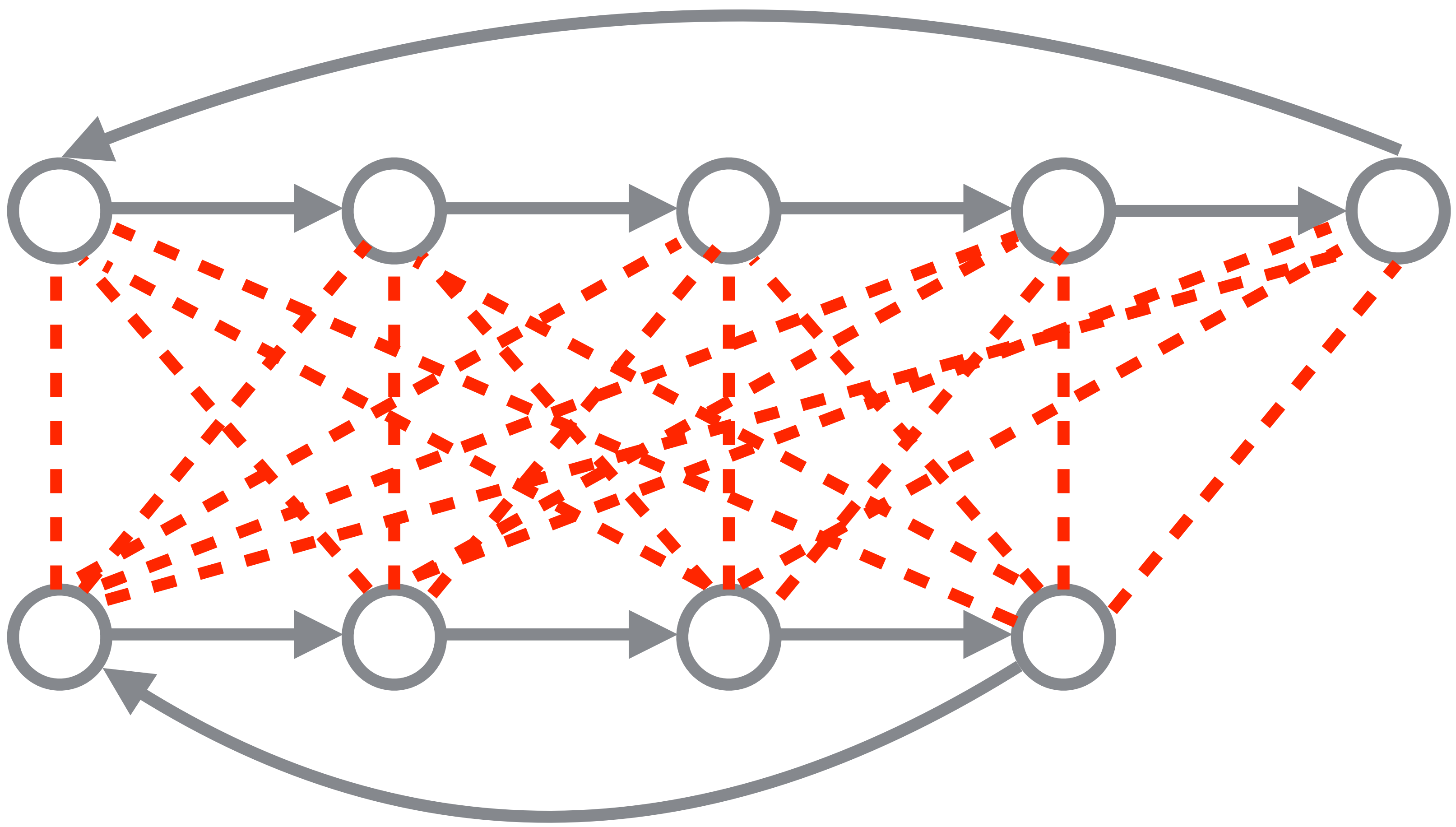
Naive Algorithm



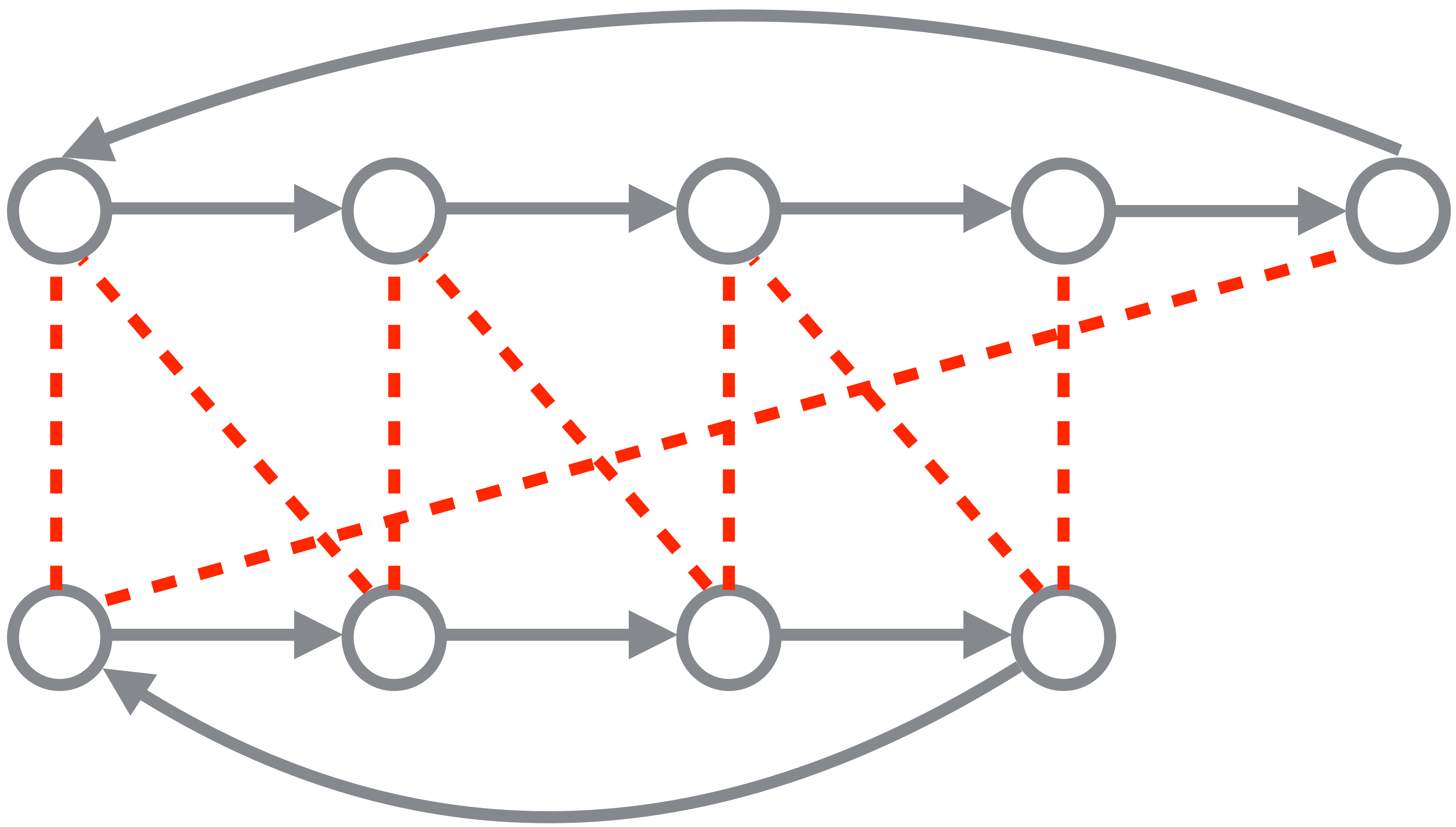
Naive Algorithm



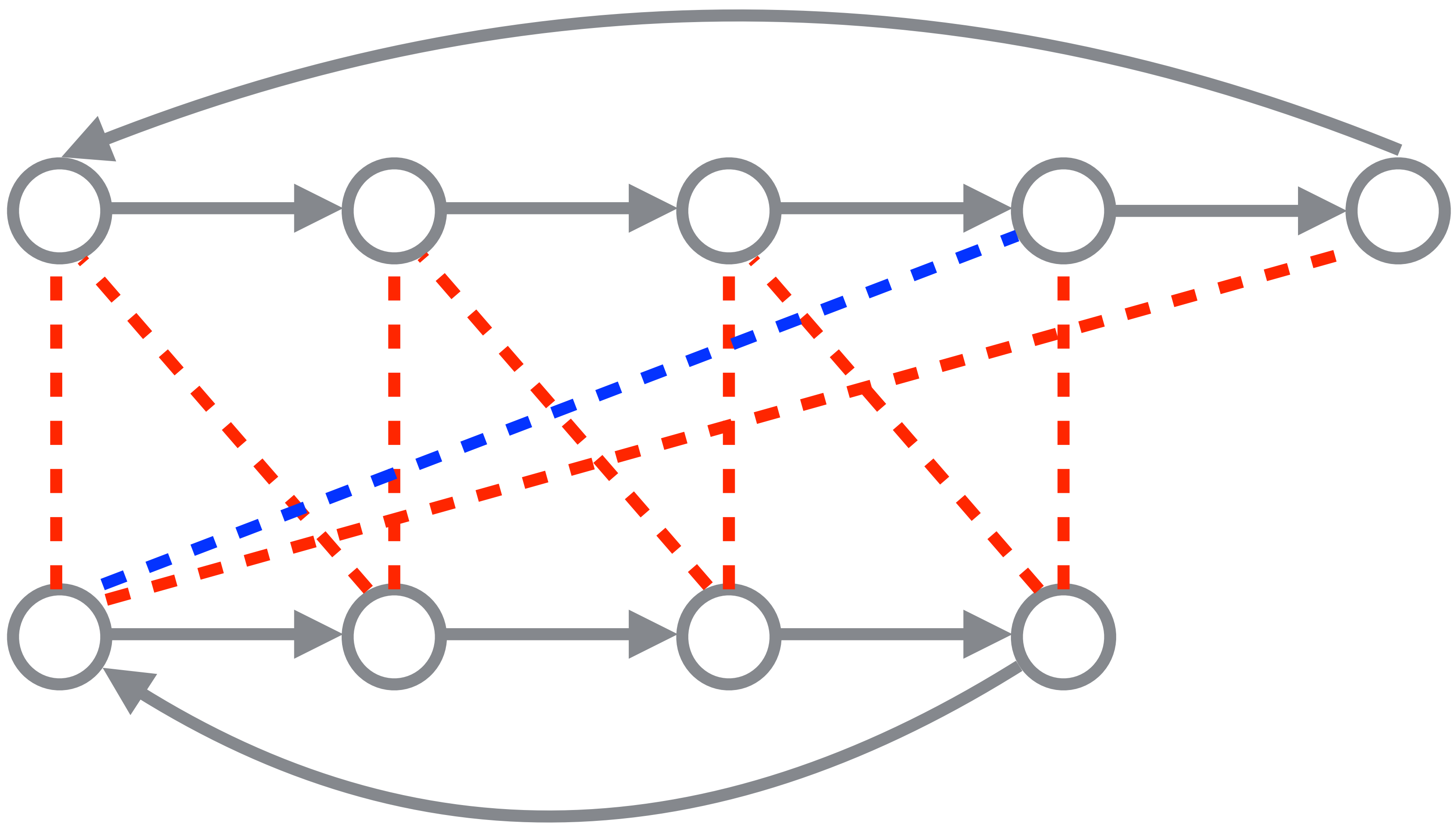
Naive Algorithm



Naive Algorithm

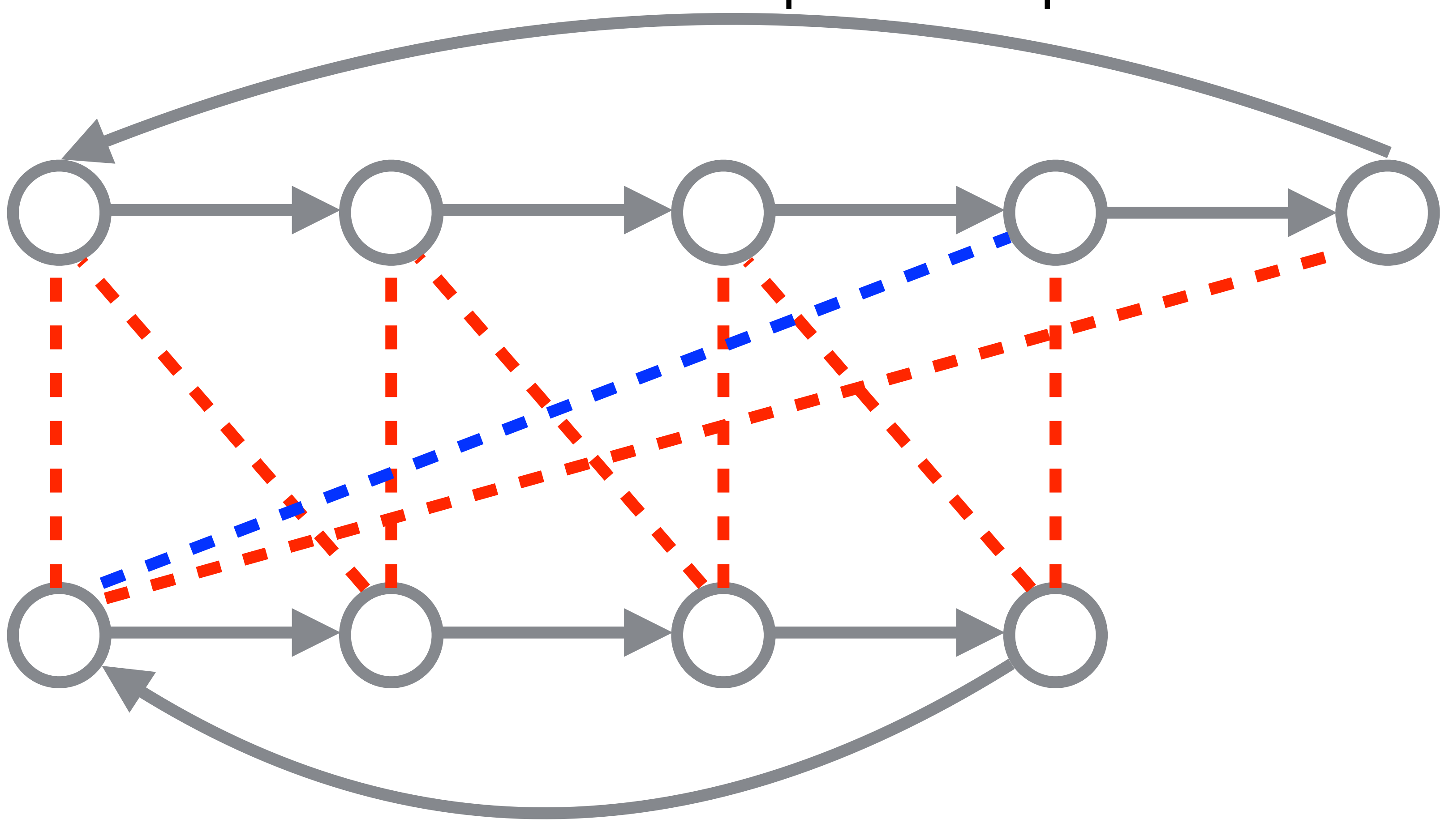


Naive Algorithm



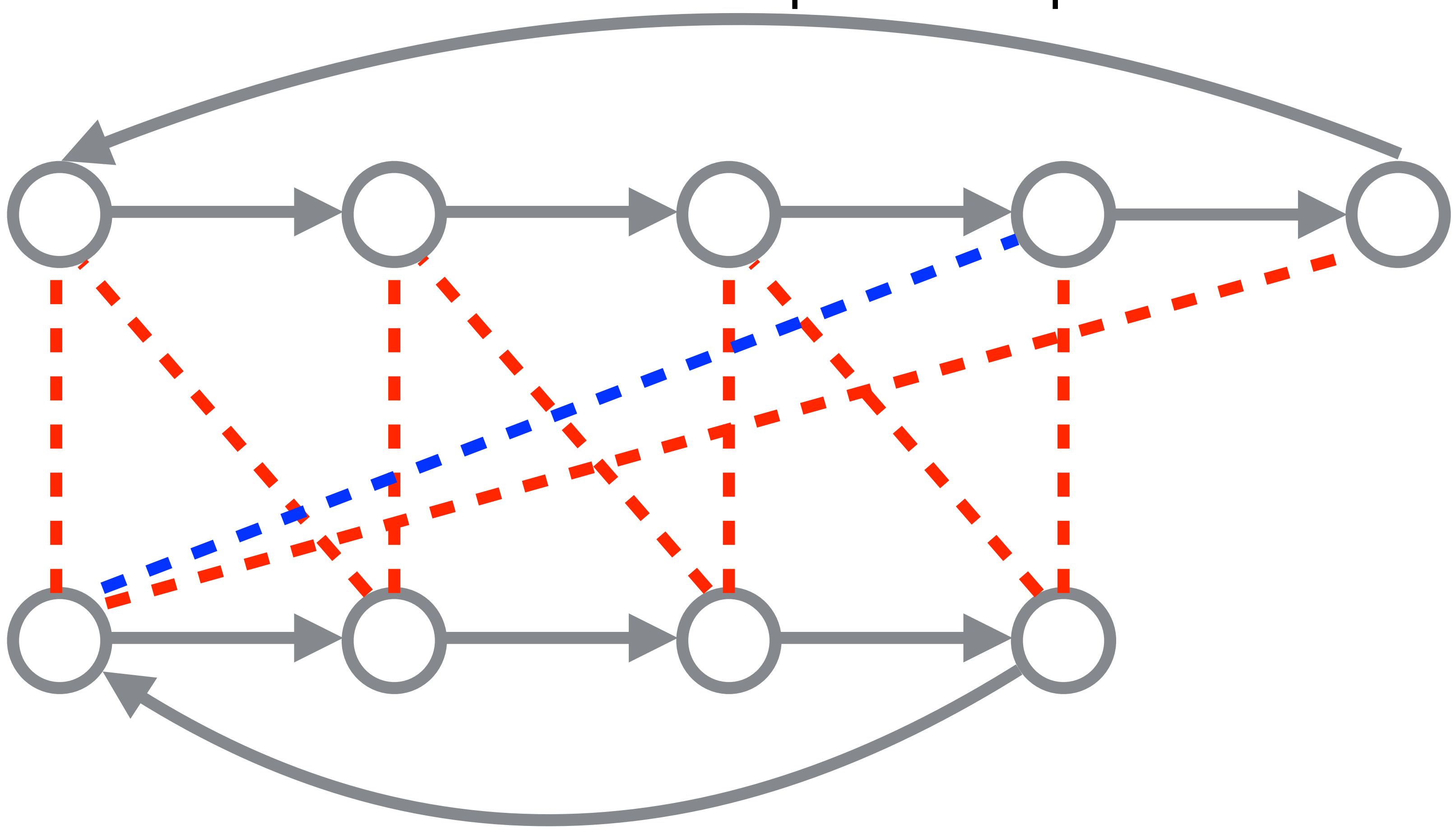
Naive Algorithm

R is not a bisimulation,
but a bisimulation up-to equivalence



Naive Algorithm

R is not a bisimulation,
but a bisimulation up-to equivalence



Eqv: **Rel$_X$**-->**Rel$_X$**

R $\subseteq$B(Eqv(R))

Hopcroft and Karp algorithm

Hopcroft and Karp algorithm

- In the worst case, the naive algorithm explores n^2 pairs

Hopcroft and Karp algorithm

- In the worst case, the naive algorithm explores n^2 pairs
- The Hopcroft and Karp algorithm (1971) builds a bisimulation up-to equivalence: it visits at most n pairs. The complexity is thus (almost) linear.

Hopcroft and Karp algorithm

- In the worst case, the naive algorithm explores n^2 pairs
- The Hopcroft and Karp algorithm (1971) builds a bisimulation up-to equivalence: it visits at most n pairs. The complexity is thus (almost) linear.
- For Non-Deterministic automata, there are smarter up-to techniques which allow for an exponential speed up

Hopcroft and Karp algorithm

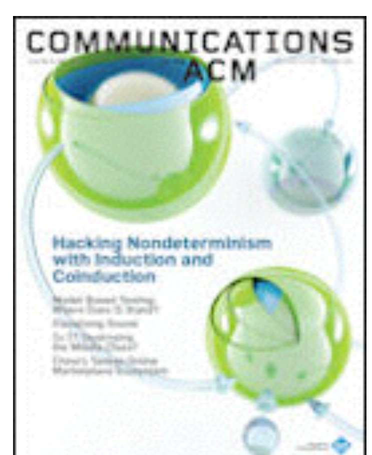
- In the worst case, the naive algorithm explores n^2 pairs
- The Hopcroft and Karp algorithm (1971) builds a bisimulation up-to equivalence: it visits at most n pairs. The complexity is thus (almost) linear.
- For Non-Deterministic automata, there are smarter up-to techniques which allow for an exponential speed up

Hopcroft and Karp algorithm

- In the worst case, the naive algorithm explores n^2 pairs
- The Hopcroft and Karp algorithm (1971) builds a bisimulation up-to equivalence: it visits at most n pairs. The complexity is thus (almost) linear.
- For Non-Deterministic automata, there are smarter up-to techniques which allow for an exponential speed up

Hopcroft and Karp algorithm

- In the worst case, the naive algorithm explores n^2 pairs
- The Hopcroft and Karp algorithm (1971) builds a bisimulation up-to equivalence: it visits at most n pairs. The complexity is thus (almost) linear.
- For Non-Deterministic automata, there are smarter up-to techniques which allow for an exponential speed up



Regular Expressions

e::= 0, 1, a, e+e, ee, e*

Brzozowski derivatives defines a DA (RE,o,t)

Regular Expressions

e::= 0, 1, a, e+e, ee, e*

Brzozowski derivatives defines a DA (RE,o,t)

$$\frac{}{1\downarrow} \qquad \frac{e\downarrow}{e+f\downarrow} \qquad \frac{f\downarrow}{e+f\downarrow} \qquad \frac{e\downarrow \quad f\downarrow}{ef\downarrow} \qquad \frac{}{e^*\downarrow}$$

Regular Expressions

$e ::= 0, 1, a, e+e, ee, e^*$

Brzozowski derivatives defines a DA (RE,o,t)

$$\frac{}{1\downarrow} \qquad \frac{e\downarrow}{e+f\downarrow} \qquad \frac{f\downarrow}{e+f\downarrow} \qquad \frac{e\downarrow \quad f\downarrow}{ef\downarrow} \qquad \frac{}{e^*\downarrow}$$

$$\frac{}{0 \xrightarrow{a} 0} \qquad \frac{}{1 \xrightarrow{a} 0} \qquad \frac{}{a \xrightarrow{a} 1} \qquad \frac{b \neq a}{b \xrightarrow{a} 1} \qquad \frac{e \xrightarrow{a} e' \quad f \xrightarrow{a} f'}{e+f \xrightarrow{a} e'+f'}$$

$$\frac{e \xrightarrow{a} e' \quad f \xrightarrow{a} f'}{ef \xrightarrow{a} e'f+o(e)f'} \qquad \frac{e \xrightarrow{a} e'}{e^* \xrightarrow{a} e'e^*}$$

Kleene Algebra

We can prove the soundness
of Kleene Algebra Axiomatization
by mean of coinduction

Commutativity: $e+f \sim f+e$

Kleene Algebra

We can prove the soundness
of Kleene Algebra Axiomatization
by mean of coinduction

Commutativity: $e+f \sim f+e$

$R = \{ (e+f, f+e) \mid e,f \in RE \}$ is a bisimulation:

Kleene Algebra

We can prove the soundness
of Kleene Algebra Axiomatization
by mean of coinduction

Commutativity: $e+f \sim f+e$

$R = \{ (e+f, f+e) \mid e,f \in RE \}$ is a bisimulation:

1. $e+f\downarrow \iff e\downarrow$ or $f\downarrow \iff f+e\downarrow$

$$\frac{e\downarrow}{e+f\downarrow} \qquad \frac{f\downarrow}{e+f\downarrow}$$

Kleene Algebra

We can prove the soundness
of Kleene Algebra Axiomatization
by mean of coinduction

Commutativity: $e+f \sim f+e$

$R=\{ (e+f, f+e) \mid e,f \in RE \}$ is a bisimulation:

1. $e+f\downarrow \iff e\downarrow$ or $f\downarrow \iff f+e\downarrow$
2. $e+f \;\; R \;\; f+e$

$$\begin{array}{ccc} e+f & R & f+e \\ \downarrow a & & \downarrow a \\ e'+f' & R & f'+e' \end{array}$$

$$\frac{e\downarrow}{e+f\downarrow} \qquad \frac{f\downarrow}{e+f\downarrow}$$

$$\frac{e \xrightarrow{a} e' \quad f \xrightarrow{a} f'}{e+f \xrightarrow{a} e'+f'}$$

Kleene Algebra

We can prove the soundness
of Kleene Algebra Axiomatization
by mean of coinduction

in a similar way, we can prove
that (RE,+,0) is a monoid

R={ (e+f,f+e) | e,f $\in$ RE } is a bisimulation:

1. $e+f\downarrow \Longleftrightarrow e\downarrow$ or $f\downarrow \Longleftrightarrow f+e\downarrow$
2. $e+f \;\; R \;\; f+e$

$$\begin{array}{ccc} e+f & R & f+e \\ \downarrow{\scriptstyle a} & & \downarrow{\scriptstyle a} \\ e'+f' & R & f'+e' \end{array}$$

$$\frac{e\downarrow}{e+f\downarrow} \qquad \frac{f\downarrow}{e+f\downarrow}$$

$$\frac{e \xrightarrow{a} e' \quad f \xrightarrow{a} f'}{e+f \xrightarrow{a} e'+f'}$$

Kleene Algebra

Distributivity: $e(f+g) \sim ef+eg$

Kleene Algebra

Distributivity: $e(f+g) \sim ef+eg$

$R = \{ (e(f+g), ef+eg) \mid e,f,g \in RE \}$ is a bisimulation:

Kleene Algebra

Distributivity: $e(f+g) \sim ef+eg$

R={ (e(f+g) ,ef+eg) | e,f,g ∈ RE } is a bisimulation:

1. $e(f+g)\downarrow \iff ef+eg\downarrow$

Kleene Algebra

Distributivity: $e(f+g) \sim ef+eg$

$R = \{ (e(f+g), ef+eg) \mid e,f,g \in RE \}$ is a bisimulation:

1. $e(f+g)\downarrow \Longleftrightarrow ef+eg\downarrow$
2. $e(f+g) \;\; R \;\; ef+eg$

$$e(f+g) \xrightarrow{a} e'(f+g)+o(e)(f'+g') \qquad ef+eg \xrightarrow{a} (e'f+o(e)f')+(e'g+o(e)g')$$

$$\frac{e \xrightarrow{a} e' \quad f \xrightarrow{a} f'}{ef \xrightarrow{a} e'f+o(e)f'}$$

$$\frac{e \xrightarrow{a} e' \quad f \xrightarrow{a} f'}{e+f \xrightarrow{a} e'+f'}$$

Kleene Algebra

Distributivity: $e(f+g) \sim ef+eg$

$R = \{ (e(f+g), ef+eg) \mid e,f,g \in RE \}$ is a bisimulation:

1. $e(f+g)\downarrow \Longleftrightarrow ef+eg\downarrow$
2. $$\begin{array}{ccc} e(f+g) & R & ef+eg \\ \downarrow a & & \downarrow a \\ e'(f+g)+o(e)(f'+g') & \not\mathrel{R} & (e'f+o(e)f')+(e'g+o(e)g') \end{array}$$

$$\frac{e \xrightarrow{a} e' \quad f \xrightarrow{a} f'}{ef \xrightarrow{a} e'f+o(e)f'}$$

$$\frac{e \xrightarrow{a} e' \quad f \xrightarrow{a} f'}{e+f \xrightarrow{a} e'+f'}$$

Kleene Algebra

Distributivity: $e(f+g) \sim ef+eg$

$R = \{ (e(f+g), ef+eg) \mid e,f,g \in RE \}$ is a bisimulation:

1. $e(f+g)\downarrow \Longleftrightarrow ef+eg\downarrow$
2. $e(f+g) \; R \; ef+eg$

$$\begin{array}{ccc} e(f+g) & R & ef+eg \\ \downarrow a & & \downarrow a \\ e'(f+g)+o(e)(f'+g') & \not R & (e'f+o(e)f')+(e'g+o(e)g') \end{array}$$

$$\frac{e \xrightarrow{a} e' \quad f \xrightarrow{a} f'}{ef \xrightarrow{a} e'f+o(e)f'}$$

$$\frac{e \xrightarrow{a} e' \quad f \xrightarrow{a} f'}{e+f \xrightarrow{a} e'+f'}$$

R is NOT a bisimulation,
but a bisimulation up to $Bhv \circ Ctx$

Kleene Algebra

Bhv: $\mathbf{Rel}_{\mathbf{RE}}$-->$\mathbf{Rel}_{\mathbf{RE}}$

Bhv(R)= { (e,f) | e∼e' R f'∼f }

Kleene Algebra

Bhv: $\mathbf{Rel}_{\mathbf{RE}} \dashrightarrow \mathbf{Rel}_{\mathbf{RE}}$

Bhv(R)= { (e,f) | e∼e' R f'∼f }

Ctx: $\mathbf{Rel}_{\mathbf{RE}} \dashrightarrow \mathbf{Rel}_{\mathbf{RE}}$

$$\frac{e \; R \; f}{e \; Ctx(R) \; f} \qquad \frac{}{0 \; Ctx(R) \; 0} \qquad \frac{}{1 \; Ctx(R) \; 1} \qquad \frac{}{a \; Ctx(R) \; a}$$

$$\frac{e \; Ctx(R) \; e' \quad f \; Ctx(R) \; f'}{e+f \; Ctx(R) \; e'+f'} \qquad \frac{e \; Ctx(R) \; e' \quad f \; Ctx(R) \; f'}{ef \; Ctx(R) \; e'f'}$$

$$\frac{e \; Ctx(R) \; f}{e^* \; Ctx(R) \; f^*}$$

Kleene Algebra

$R = \{ (e(f+g), ef+eg) \mid e,f,g \in RE \}$

$e(f+g) \quad R \quad ef+eg$

$\downarrow a \qquad\qquad \downarrow a$

$e'(f+g)+o(e)(f'+g') \quad Bhv(Ctx(R)) \quad (e'f+o(e)f')+(e'g+o(e)g')$

R is NOT a bisimulation,
but a bisimulation up to Bhv∘Ctx

Kleene Algebra

$$R = \{ (e(f+g), ef+eg) \mid e,f,g \in RE \}$$

$$
\begin{array}{ccc}
e(f+g) & R & ef+eg \\
\downarrow a & & \downarrow a \\
e'(f+g)+o(e)(f'+g') & Bhv(Ctx(R)) & (e'f+o(e)f')+(e'g+o(e)g')
\end{array}
$$

$$e'(f+g)+o(e)(f'+g') \; Ctx(R) \; (e'f+e'g)+(o(e)f'+o(e)g') \sim (e'f+o(e)f')+(e'g+o(e)g')$$

R is NOT a bisimulation,
but a bisimulation up to $Bhv \circ Ctx$

Kleene Algebra

$R=\{ (e(f+g), ef+eg) \mid e,f,g \in RE \}$

$e(f+g) \quad R \quad ef+eg$

$\downarrow a \qquad\qquad \downarrow a$

$e'(f+g)+o(e)(f'+g') \quad Bhv(Ctx(R)) \quad (e'f+o(e)f')+(e'g+o(e)g')$

$e'(f+g)+o(e)(f'+g') \; Ctx(R) \; (e'f+e'g)+(o(e)f'+o(e)g') \sim (e'f+o(e)f')+(e'g+o(e)g')$

$R \subseteq B(Bhv(Ctx(R)))$

R is NOT a bisimulation,
but a bisimulation up to Bhv∘Ctx

Arden's rule

Given two regular expressions k and m, the equation

$$e \sim ke+m$$

has solution $e=k^*m$, i.e., $k^*m \sim kk^*m + m$

Moreover:

1. $k\not\downarrow \Rightarrow k^*m$ is the *unique* solution, i.e., $f \sim kf+m \Rightarrow f \sim k^*m$
2. k^*m is the *smallest* solution, i.e., $f \sim kf+m \Rightarrow k^*m \lesssim f$

Arden's rule

Given two regular expressions k and m, the equation

$$e \sim ke + m$$

has solution $e = k^*m$, i.e., $k^*m \sim kk^*m + m$

Can be proved by coinduction

Moreover:

1. $k{\not\downarrow} \Rightarrow k^*m$ is the *unique* solution, i.e., $f \sim kf + m \Rightarrow f \sim k^*m$
2. k^*m is the *smallest* solution, i.e., $f \sim kf + m \Rightarrow k^*m \lesssim f$

Arden's rule

Given two regular expressions k and m, the equation

$$e \sim ke+m$$

has solution $e=k^*m$, i.e., $k^*m \sim kk^*m + m$

Can be proved by coinduction

Moreover:

1. $k\not\downarrow \Rightarrow k^*m$ is the *unique* solution, i.e., $f \sim kf+m \Rightarrow f \sim k^*m$
2. k^*m is the *smallest* solution, i.e., $f \sim kf+m \Rightarrow k^*m \lesssim f$

language inclusion ($\lesssim$) is $\nu B'$

B':**Rel$_X$**-->**Rel$_X$** is defined as

$B'(R)=\{(x,y) \mid o(x) \leq o(y)$ and for all $a \in A$ $t(x)(a)\ R\ t(y)(a)\}$

Arden's rule

To show $f \sim kf+m \Rightarrow k^*m \lesssim f$

We prove that
$S = \{ (k^*m,f) \mid f \sim kf+m \}$
is a simulation up-to

Arden's rule

To show $f \sim kf+m \Rightarrow k^*m \lesssim f$

We prove that
$S = \{ (k^*m, f) \mid f \sim kf+m \}$
is a simulation up-to

1. $k^*m\downarrow \Rightarrow m\downarrow \Rightarrow kf+m\downarrow \Rightarrow f\downarrow$
2. $k^*m \xrightarrow{a}$ ... $f \xrightarrow{a}$

$(k'k^*)m + 1m' \sim k'(k^*m) + m' \; Ctx(S) \; k'f+m' \lesssim (k'f + o(k)f') + m' \sim f'$

Arden's rule

To show $f \sim kf+m \Rightarrow k^*m \lesssim f$

We prove that

$S = \{ (k^*m, f) \mid f \sim kf+m \}$

is a simulation up-to

1. $k^*m\downarrow \;\Rightarrow\; m\downarrow \;\Rightarrow\; kf+m\downarrow \;\Rightarrow f\downarrow$
2. k^*m $\xrightarrow{a}$ … f $\xrightarrow{a}$

$(k'k^*)m+1m' \sim k'(k^*m) + m' \; Ctx(S) \; k'f+m' \lesssim (k'f + o(k)f')+m' \sim f'$

$S \subseteq B'(Slf(Ctx(S)))$

Slf: $\mathbf{Rel}_{\mathbf{RE}} \text{-->} \mathbf{Rel}_{\mathbf{RE}}$

$Slf(S) = \{ (e,f) \mid e \lesssim e' \; S \; f' \lesssim f \}$

Proving Soundness

We need to prove that these techniques are sound
(they do NOT follow from Knaster-Tarski)

Proving Soundness

We need to prove that these techniques are sound
(they do NOT follow from Knaster-Tarski)

Proving soundness is rather complicated and error prone

Proving Soundness

We need to prove that these techniques are sound
(they do NOT follow from Knaster-Tarski)

Proving soundness is rather complicated and error prone

In Milner's book there are two mistakes:

Weak Bisimulation up to weak bisimilarity

Weak Bisimulation up to equivalence

Desiderata

We would like to be able to prove soundness for

- Different sort of up-to techniques (like Eqv, Bhv, Ctx, Slf)
- Different sort of coinductive predicates (like $\sim$ or $\lesssim$)
- Different sort of systems (like DA or LTS)

Moreover, we would like to prove the soundness of these techniques in a modular way:

$$\text{Ctx and Bhv are sound} \Rightarrow \text{Bhv}\circ\text{Ctx is sound}$$

The Double role of Coalgebra

	Coalgebras as Systems	**Coalgebras as Proofs**
Functor F	F: **Set**→**Set** Type of the systems	F: $\mathbf{Rel}_X \to \mathbf{Rel}_X$ Type of the Proof
F-coalgebra	System X→FX	Invariants X⊆FX
Final F-coalgebra	Universe of Behaviours	Coinductive Predicate νF

The Double role of Coalgebra

	Coalgebras as Systems	**Coalgebras as Proofs**
Functor F	F: **Set**→**Set** Type of the systems	F: $\mathbf{Rel}_X \to \mathbf{Rel}_X$ Type of the Proof
F-coalgebra	System X→FX	Invariants X⊆FX
Final F-coalgebra	Universe of Behaviours	Coinductive Predicate νF

An up-to technique is a functor G: $\mathbf{Rel}_X \to \mathbf{Rel}_X$

An F-invariant up-to G is a coalgebra X⊆FGX

The Double role of Coalgebra

An up-to technique is a functor G: $\mathbf{Rel_X} \rightarrow \mathbf{Rel_X}$

An F-invariant up-to G is a coalgebra $X \subseteq FGX$

The Double role of Coalgebra

G is *sound* if there exists a functor H:**Coalg(FG)**→**Coalg(F)**
and a natural transformation κ:U⇒UH

$$
\begin{array}{ccc}
\mathbf{Coalg(FG)} & \xrightarrow{\quad H \quad} & \mathbf{Coalg(F)} \\
U \downarrow & \Nearrow & \downarrow U \\
\mathbf{Rel_X} & \xrightarrow{\quad Id \quad} & \mathbf{Rel_X}
\end{array}
$$

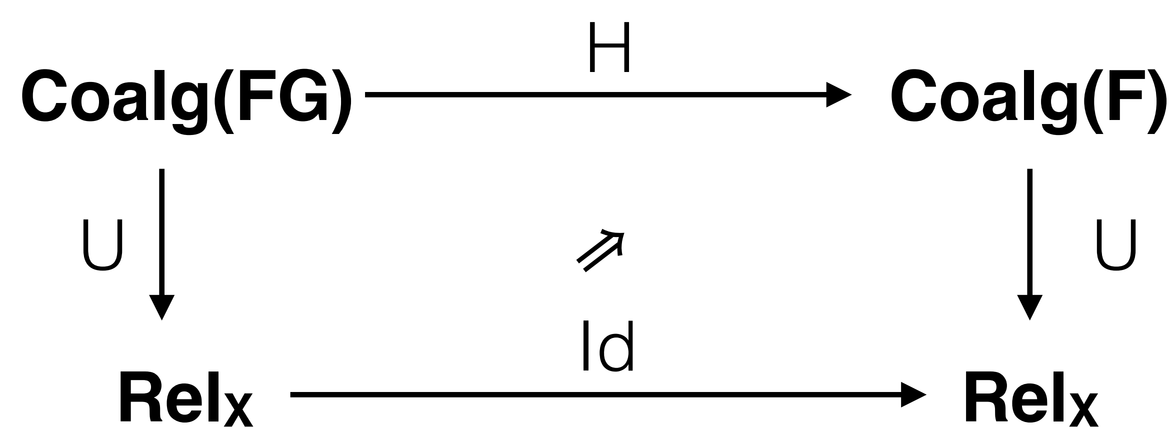
An up-to technique is a functor G: $\mathbf{Rel_X} \rightarrow \mathbf{Rel_X}$

An F-invariant up-to G is a coalgebra X⊆FGX

The Double role of Coalgebra

G is *sound* if there exists a functor H:**Coalg(FG)**→**Coalg(F)** and a natural transformation κ:U⇒UH

intuitively H transforms X⊆FGX into Y⊆FY



intuitively κ ensures that X⊆Y

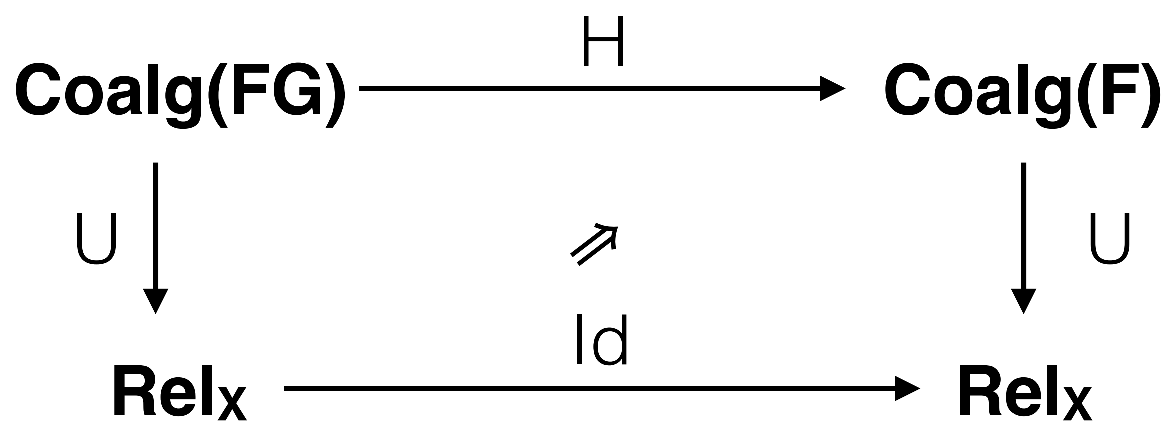
An up-to technique is a functor G: **Rel_X**→**Rel_X**

An F-invariant up-to G is a coalgebra X⊆FGX

The Double role of Coalgebra

G is *sound* if there exists a functor H:**Coalg(FG)**→**Coalg(F)**
and a natural transformation κ:U⇒UH

intuitively H transforms X⊆FGX into Y⊆FY



intuitively κ ensures that X⊆Y

∃Z, Z ⊆ X ⊆ FGX ⇒ ∃Z, Z ⊆ Y ⊆ FY ⇒ Z ⊆ νF

Generalised Powerset Construction

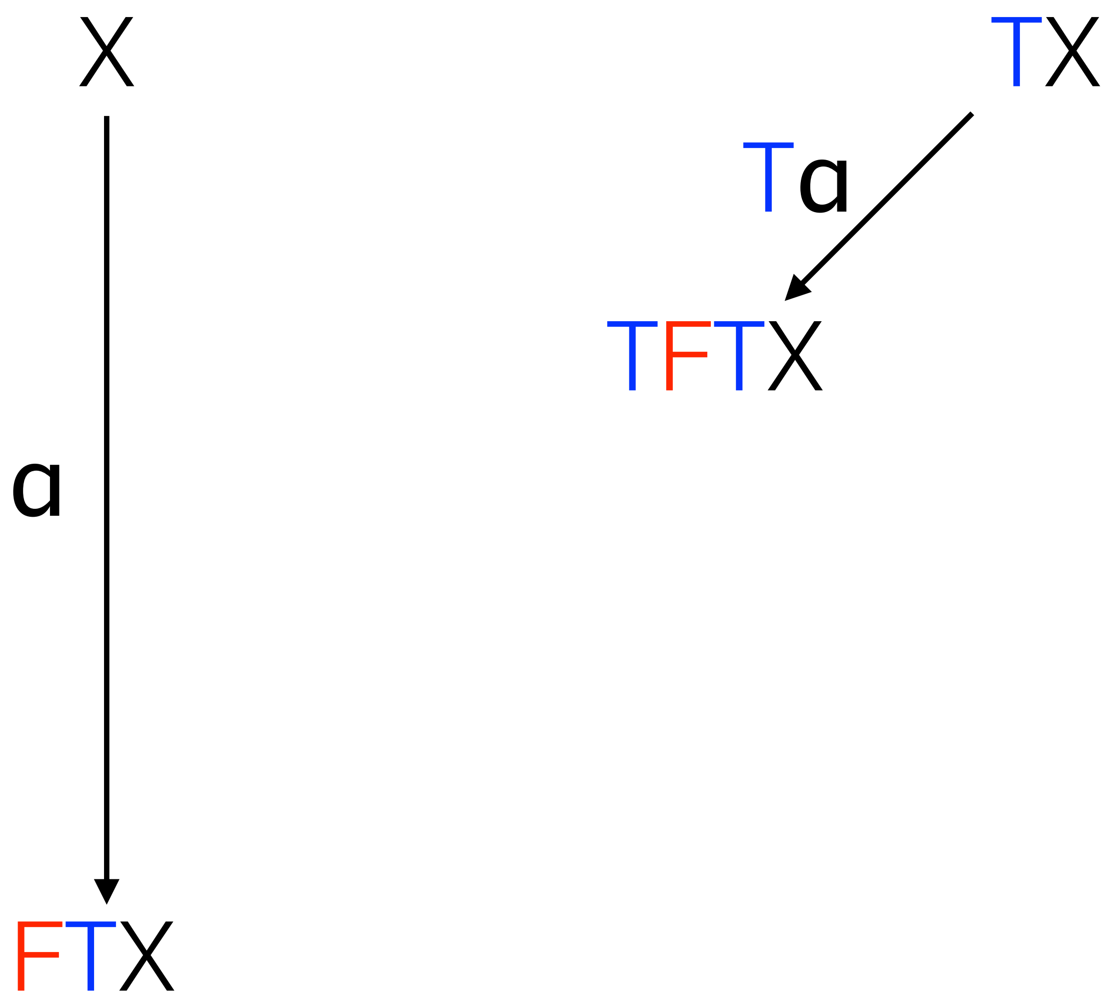
Silva, Bonchi, Bonsangue, Rutten - FSTTCS 2010

$$X \xrightarrow{\alpha} FTX$$

a functor F, a monad (T,η,μ) and a distributive law λ:TF ⇒FT

Generalised Powerset Construction

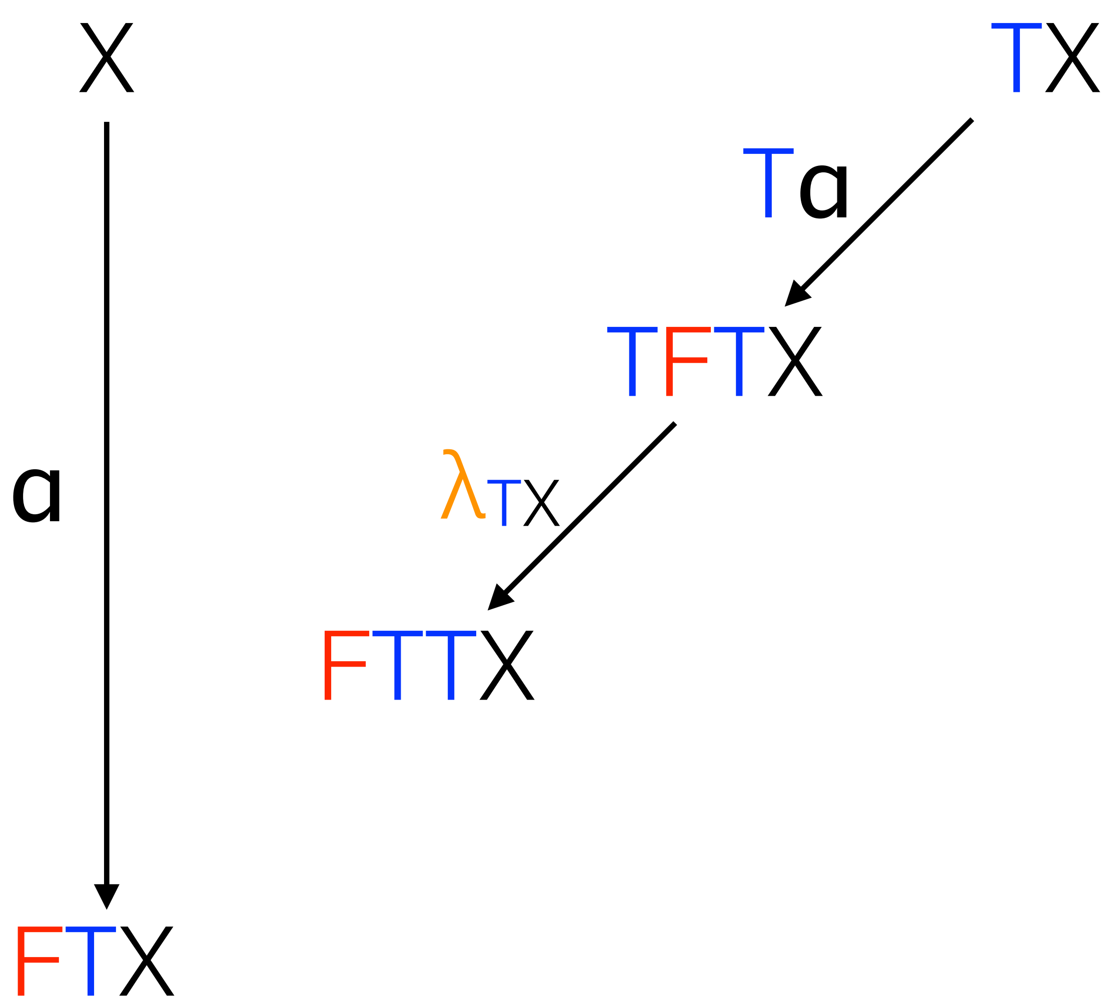
Silva, Bonchi, Bonsangue, Rutten - FSTTCS 2010



a functor F, a monad (T,η,μ) and a distributive law λ:TF ⇒FT

Generalised Powerset Construction

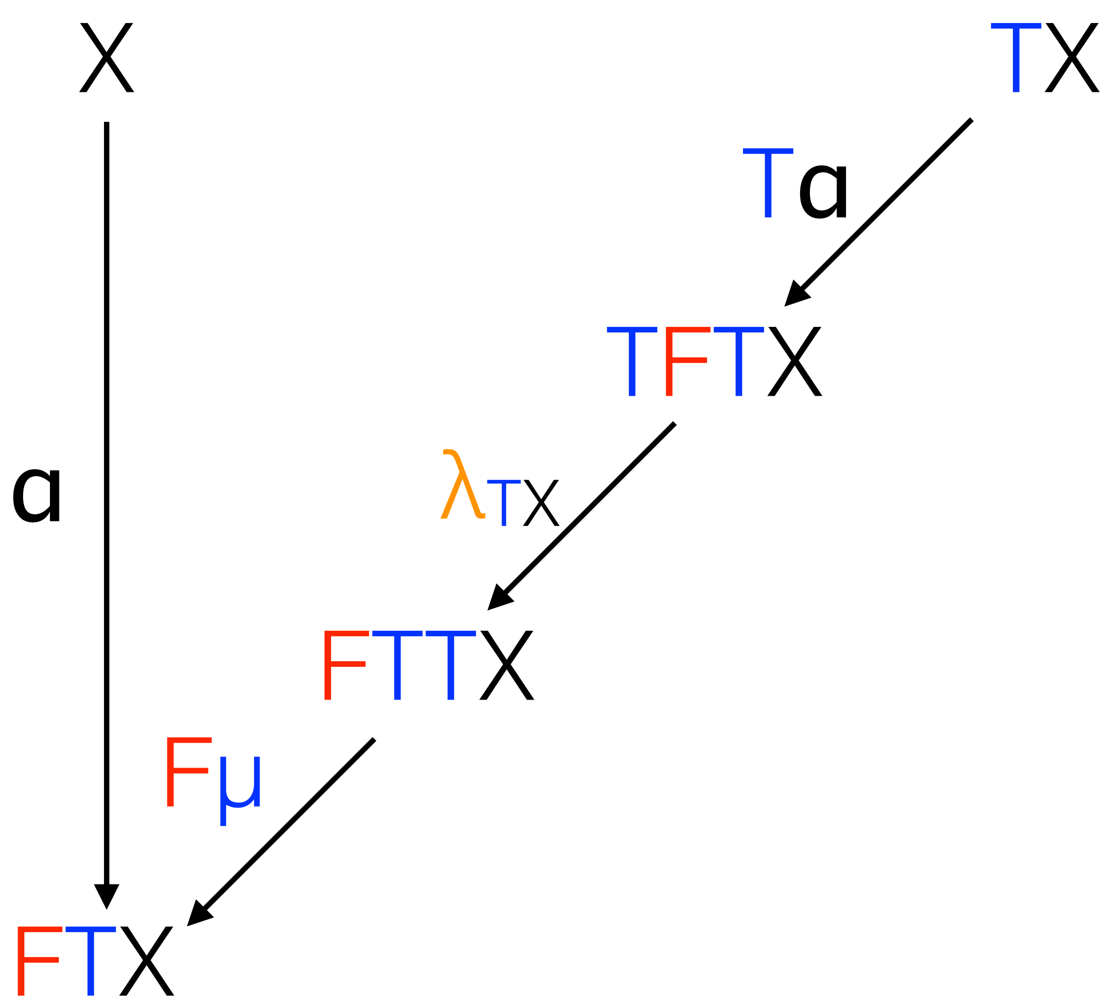
Silva, Bonchi, Bonsangue, Rutten - FSTTCS 2010



a functor F, a monad (T,η,μ) and a distributive law $\lambda : TF \Rightarrow FT$

Generalised Powerset Construction

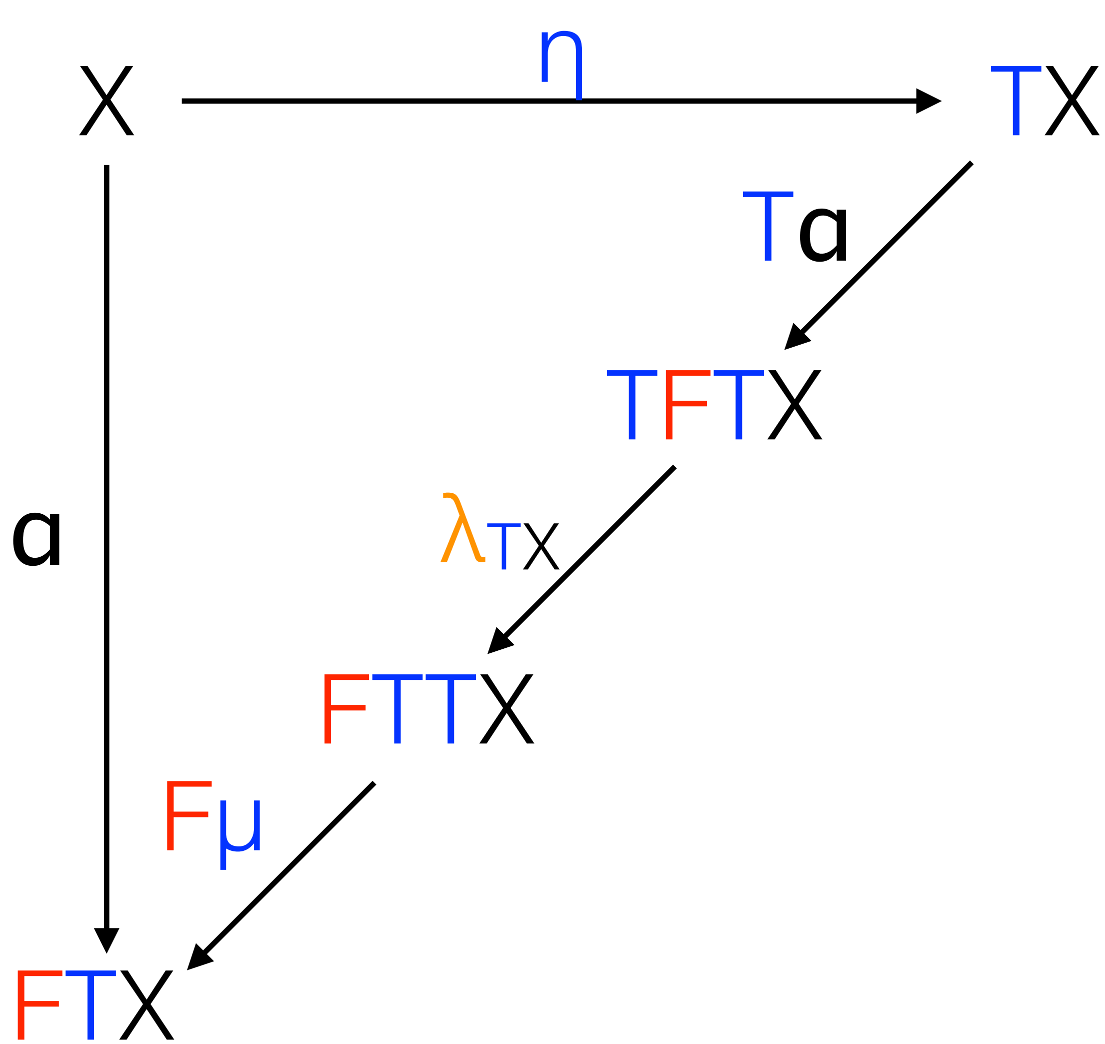
Silva, Bonchi, Bonsangue, Rutten - FSTTCS 2010



a functor F, a monad (T,η,μ) and a distributive law λ:TF ⇒FT

Generalised Powerset Construction

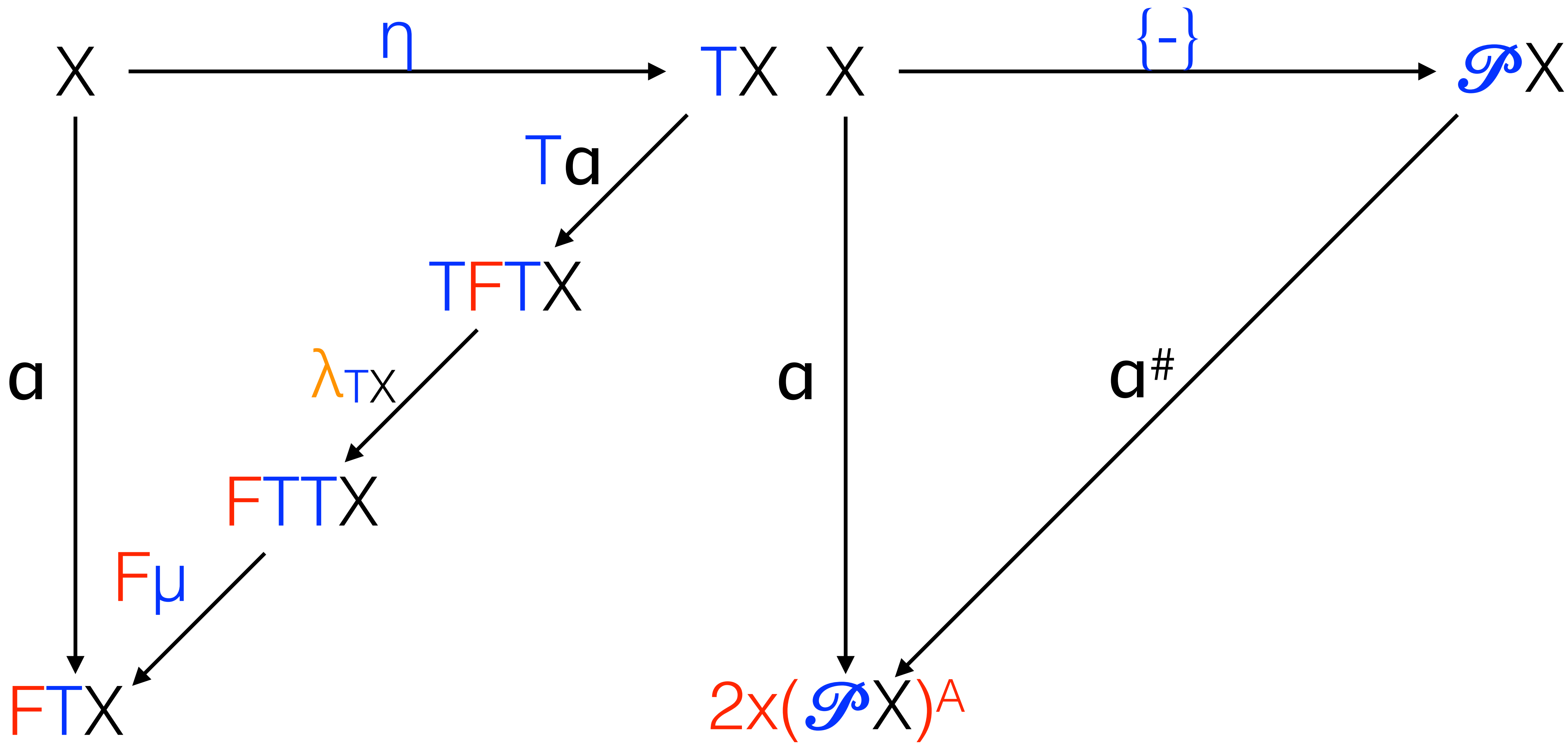
Silva, Bonchi, Bonsangue, Rutten - FSTTCS 2010



a functor F, a monad (T,η,μ) and a distributive law λ:TF ⇒FT

Generalised Powerset Construction

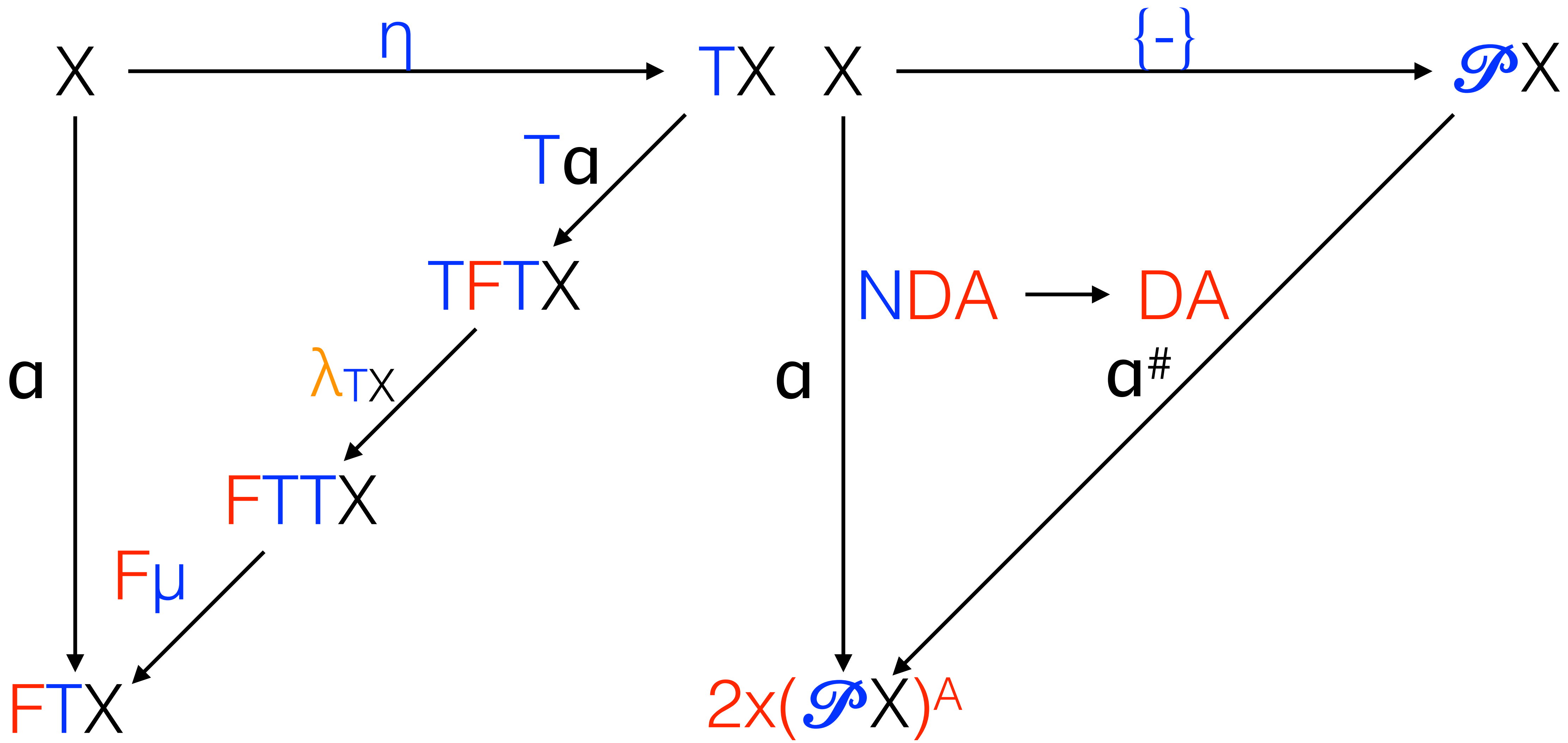
Silva, Bonchi, Bonsangue, Rutten - FSTTCS 2010



a functor F, a monad (T,η,μ) and a distributive law λ:TF ⇒FT

Generalised Powerset Construction

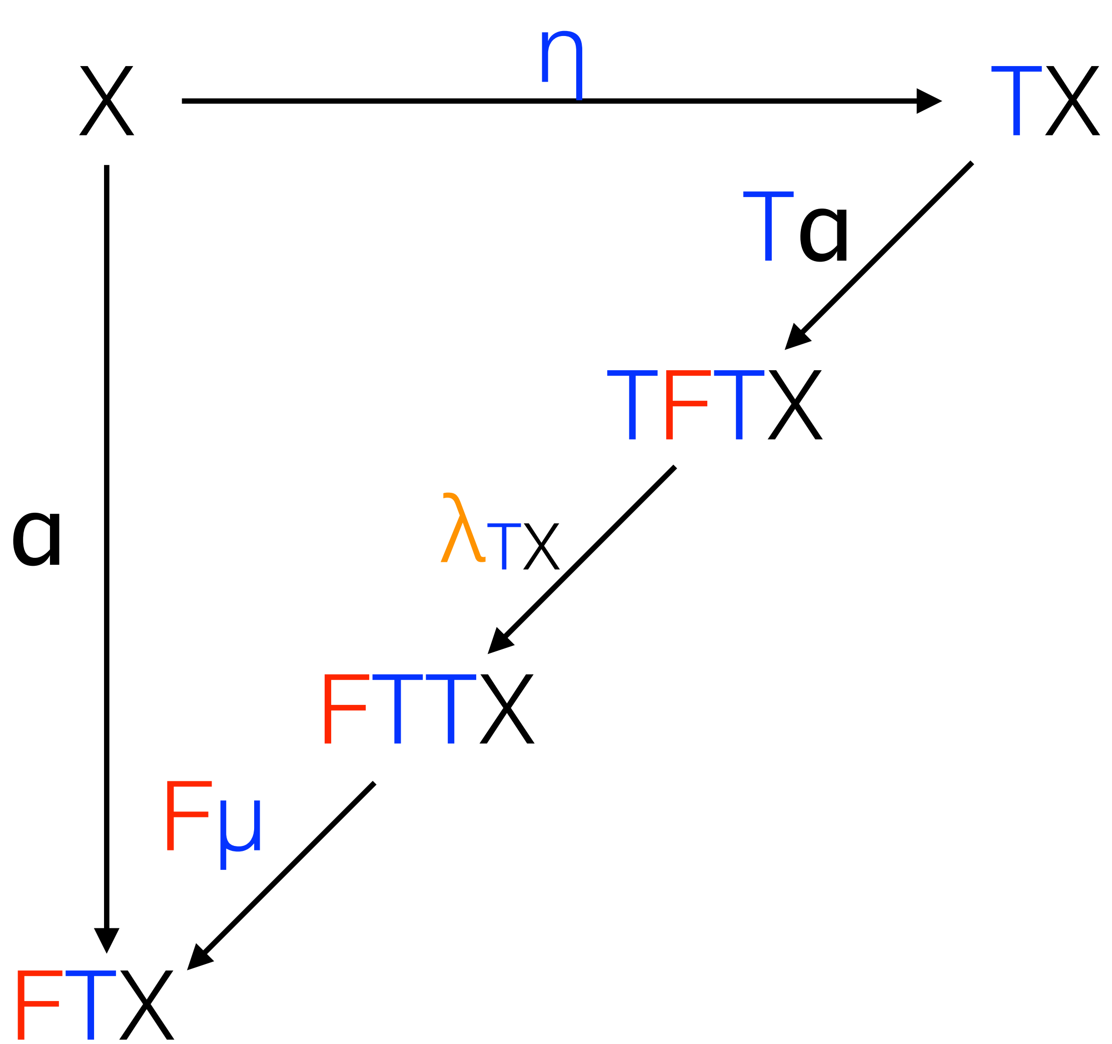
Silva, Bonchi, Bonsangue, Rutten - FSTTCS 2010



a functor F, a monad (T,η,μ) and a distributive law λ:TF ⇒FT

Generalised Powerset Construction

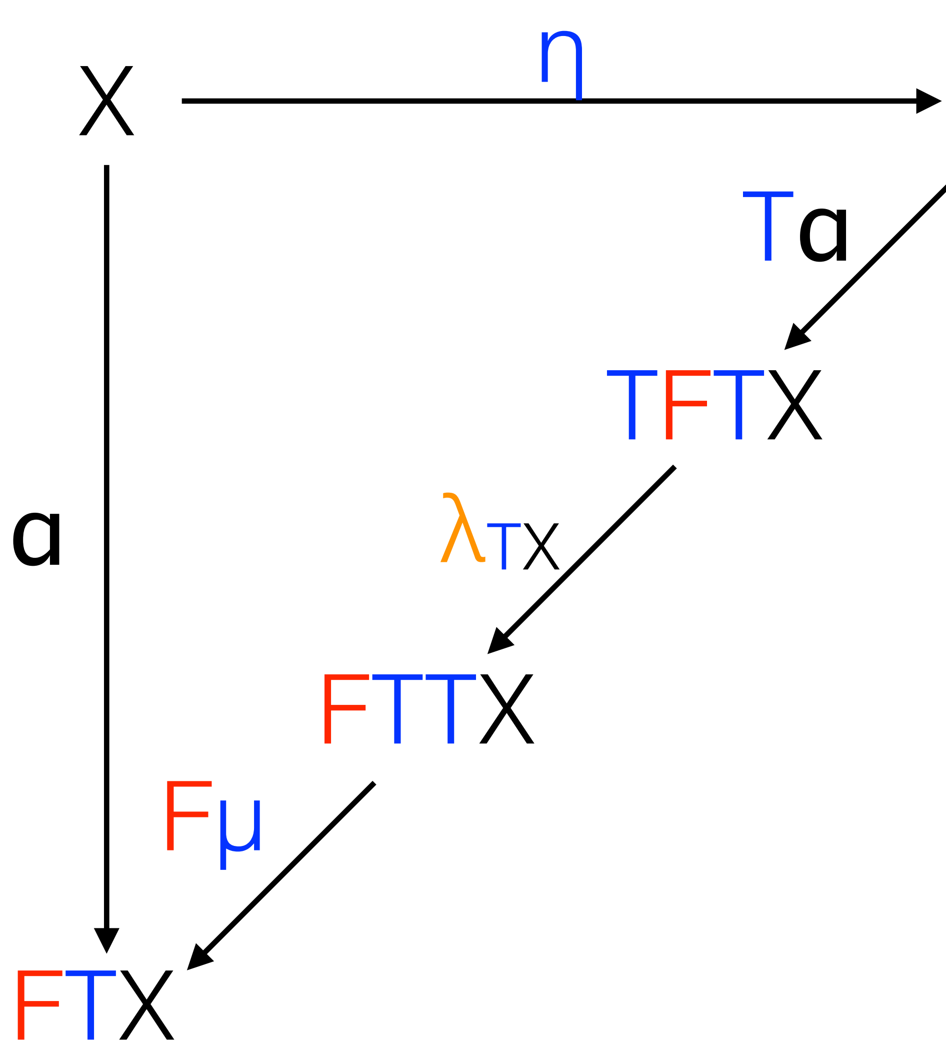
Silva, Bonchi, Bonsangue, Rutten - FSTTCS 2010



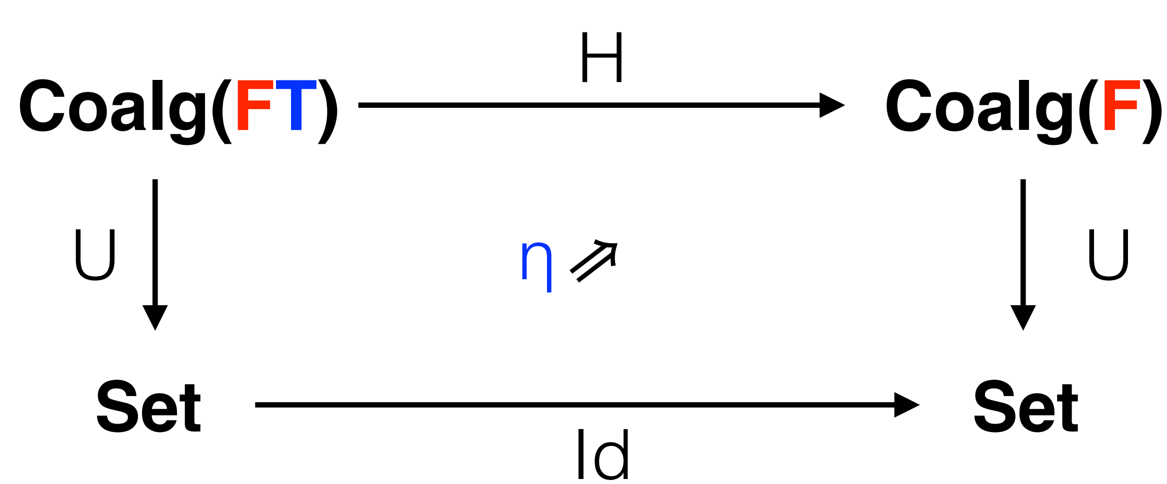
a functor F, a monad (T,η,μ) and a distributive law $\lambda : TF \Rightarrow FT$

Generalised Powerset Construction

Silva, Bonchi, Bonsangue, Rutten - FSTTCS 2010



Theorem



a functor F, a monad (T,η,μ) and a distributive law $\lambda: TF \Rightarrow FT$

The Double role of Coalgebra

	Coalgebras as Systems	**Coalgebras as Proofs**
Functor F	F: **Set**→**Set** Type of the systems	F: $\mathbf{Rel}_X \to \mathbf{Rel}_X$ Proof technique
F-coalgebra	System X→FX	Invariants X⊆FX
Final F-coalgebra	Universe of Behaviours	Coinductive Predicate νF
FT-coalgebra	F-sytem with branching T	F-Invariants up-to T

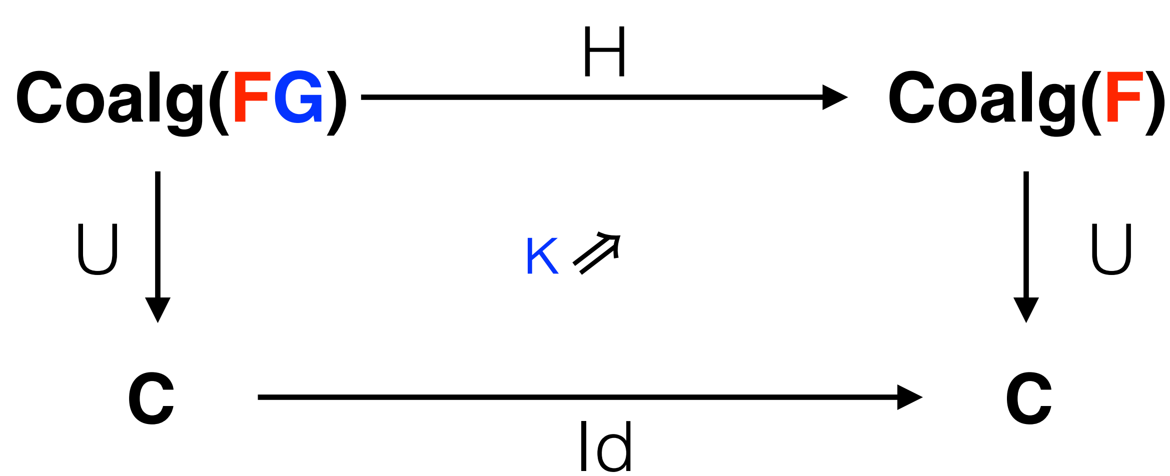
Compatible Functors

Actually, we need much less than a monad T...

Compatible Functors

Actually, we need much less than a monad T...

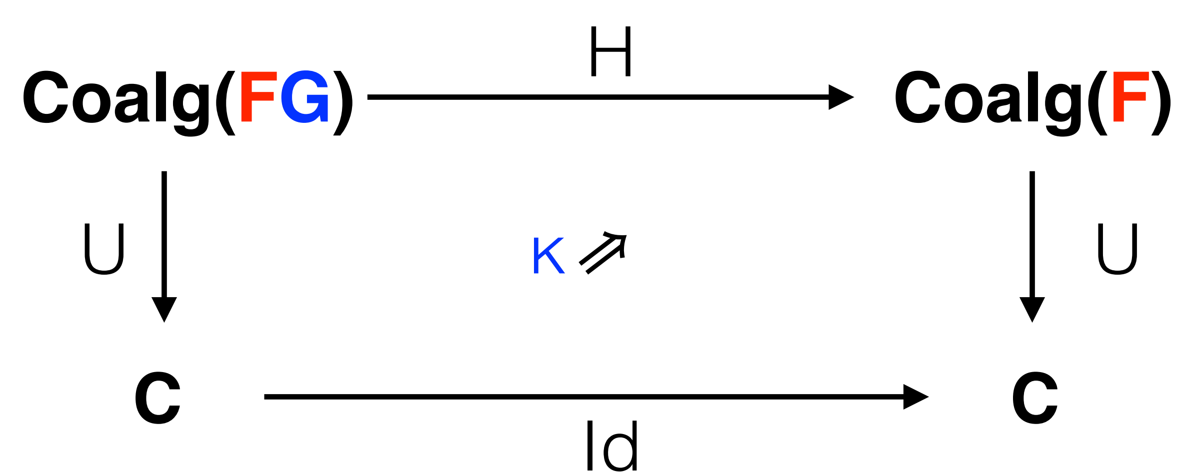
<u>Theorem</u>: a category **C** with countable coproducts F,G:**C**→**C** and λ:GF ⇒FG. Then



Compatible Functors

Actually, we need much less than a monad T...

<u>Theorem</u>: a category **C** with countable coproducts F,G:**C**→**C** and λ:GF ⇒FG. Then



G is said to be *compatible* with F iff there is λ:GF⇒FG

Compatible Functors

Actually, we need much less than a monad T...

G is said to be *compatible* with F iff there is $\lambda : GF \Rightarrow FG$

Compatible Functors

Actually, we need much less than a monad T...

Theorem: in a category **C** with countable coproducts, F-compatibility implies F-soundness

G is said to be *compatible* with F iff there is $\lambda: GF \Rightarrow FG$

Compatible Functors

Actually, we need much less than a monad T...

<u>Theorem</u>: in a category **C** with countable coproducts, F-compatibility implies F-soundness

<u>Compositionality Theorem</u>
If G_1 and G_2 are compatibile with F,
then $G_1 \circ G_2$ is compatible with F

G is said to be *compatible* with F iff there is $\lambda : GF \Rightarrow FG$

Coinductive Predicates

Hermida and Jacobs - Information and Computation 1998

Category **Rel**

objects: $R \subseteq X \times X$

arrows $R \subseteq X \times X \to S \subseteq Y \times Y$: $f: X \to Y$ such that $f(R) \subseteq f(S)$

Coinductive Predicates

Hermida and Jacobs - Information and Computation 1998

Category **Rel**

objects: $R \subseteq X \times X$

arrows $R \subseteq X \times X \to S \subseteq Y \times Y$: $f: X \to Y$ such that $f(R) \subseteq f(S)$

$$\mathbf{Rel} \xrightarrow{\;p\;} \mathbf{Set}$$

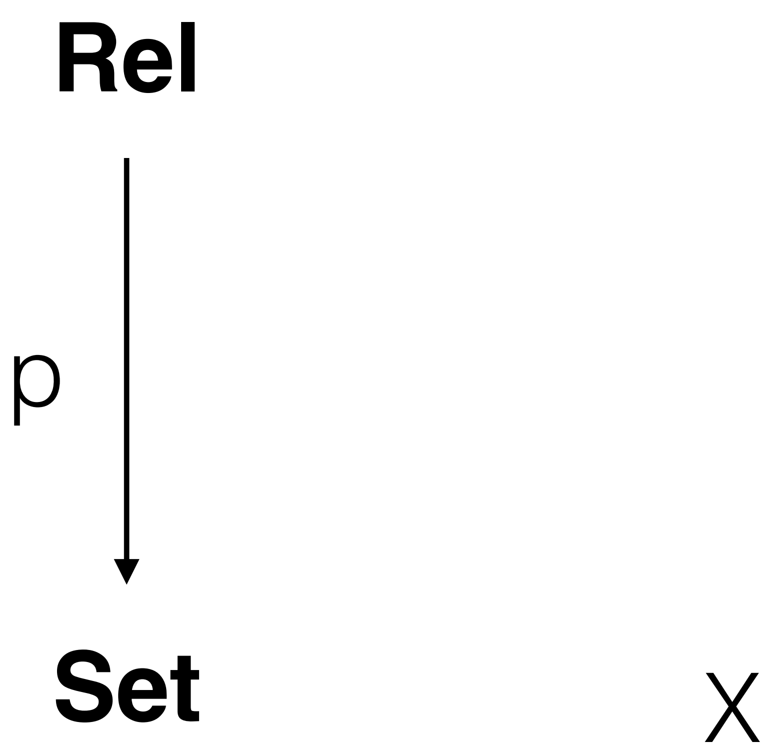
Coinductive Predicates

Hermida and Jacobs - Information and Computation 1998

Category **Rel**

objects: R⊆XxX

arrows R⊆XxX→S⊆YxY: f:X→Y such that f(R)⊆f(S)



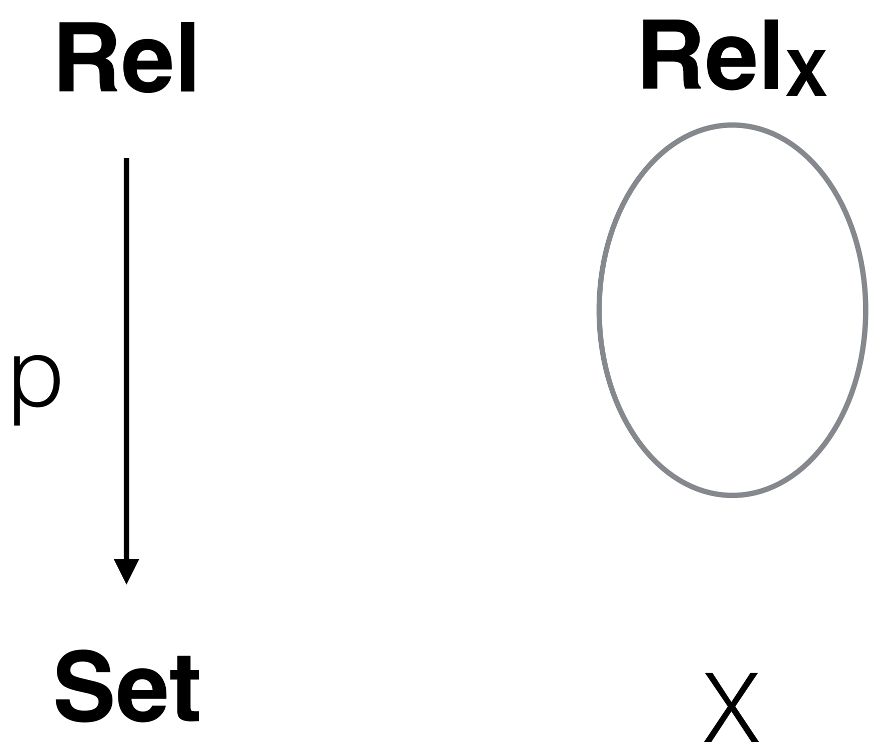
Coinductive Predicates

Hermida and Jacobs - Information and Computation 1998

Category **Rel**

objects: R⊆XxX

arrows R⊆XxX→S⊆YxY: f:X→Y such that f(R)⊆f(S)



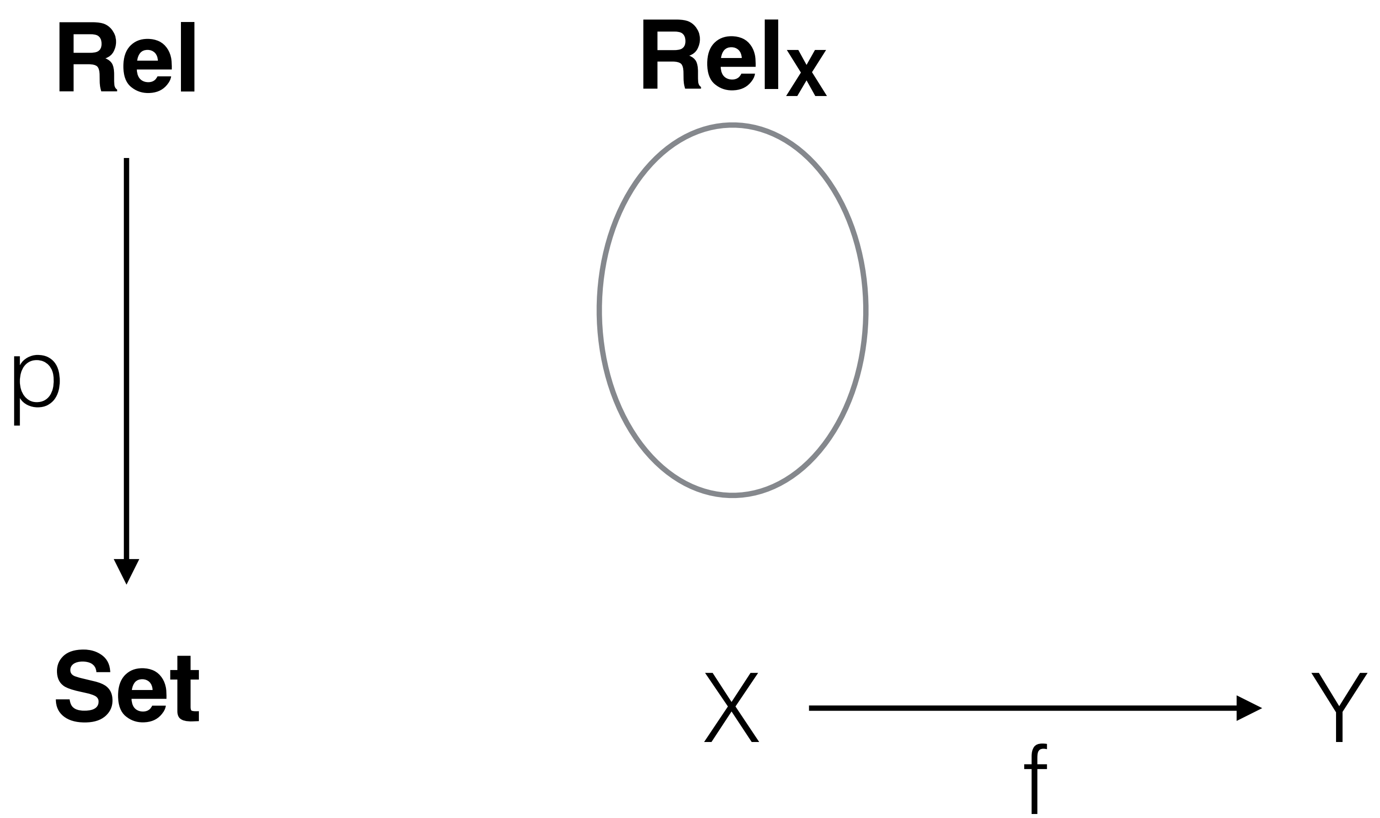
Coinductive Predicates

Hermida and Jacobs - Information and Computation 1998

Category **Rel**

objects: R⊆XxX

arrows R⊆XxX→S⊆YxY: f:X→Y such that f(R)⊆f(S)



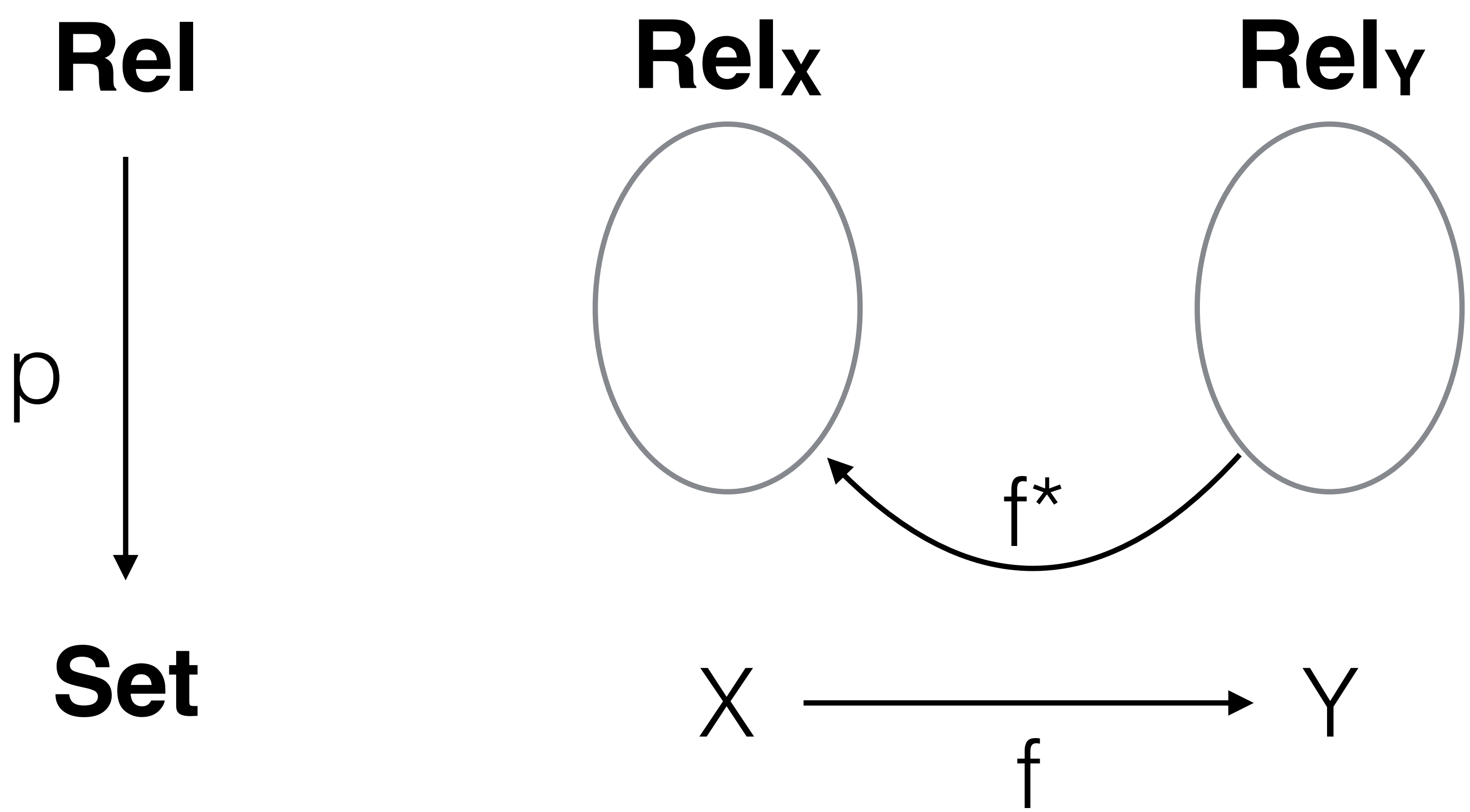
Coinductive Predicates

Hermida and Jacobs - Information and Computation 1998

Category **Rel**

objects: $R \subseteq X \times X$

arrows $R \subseteq X \times X \to S \subseteq Y \times Y$: $f: X \to Y$ such that $f(R) \subseteq f(S)$



$$f^*(S) = \{ (x,y) \mid f(x)\ S\ f(y) \}$$

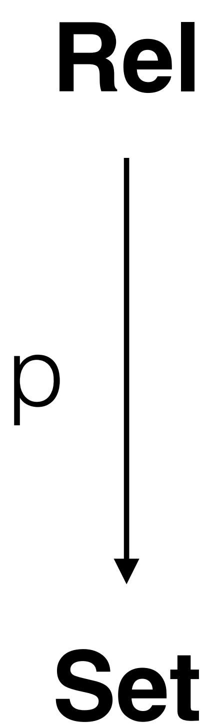
Coinductive Predicates

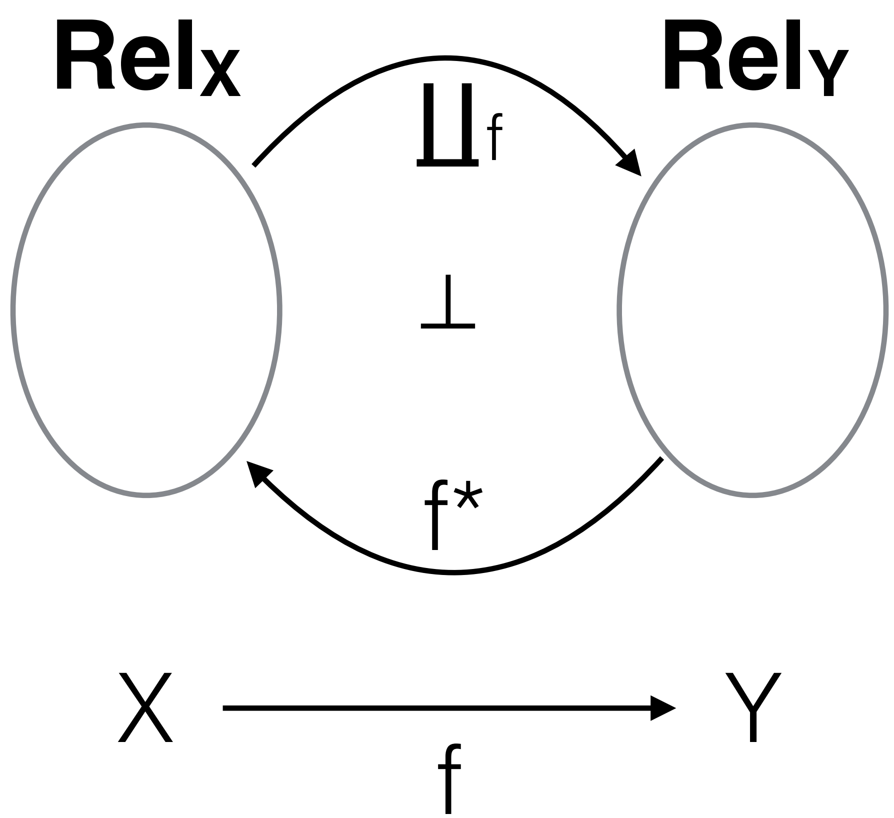
Hermida and Jacobs - Information and Computation 1998

Category **Rel**

objects: R⊆XxX

arrows R⊆XxX→S⊆YxY: f:X→Y such that f(R)⊆f(S)





$\coprod_f(R)$
$=$
$\{ (f(x),f(y)) \mid x \; R \; y\}$

$f^*(S)$
$=$
$\{ (x,y) \mid f(x) \; S \; f(y)\}$

Coinductive Predicates

Hermida and Jacobs - Information and Computation 1998

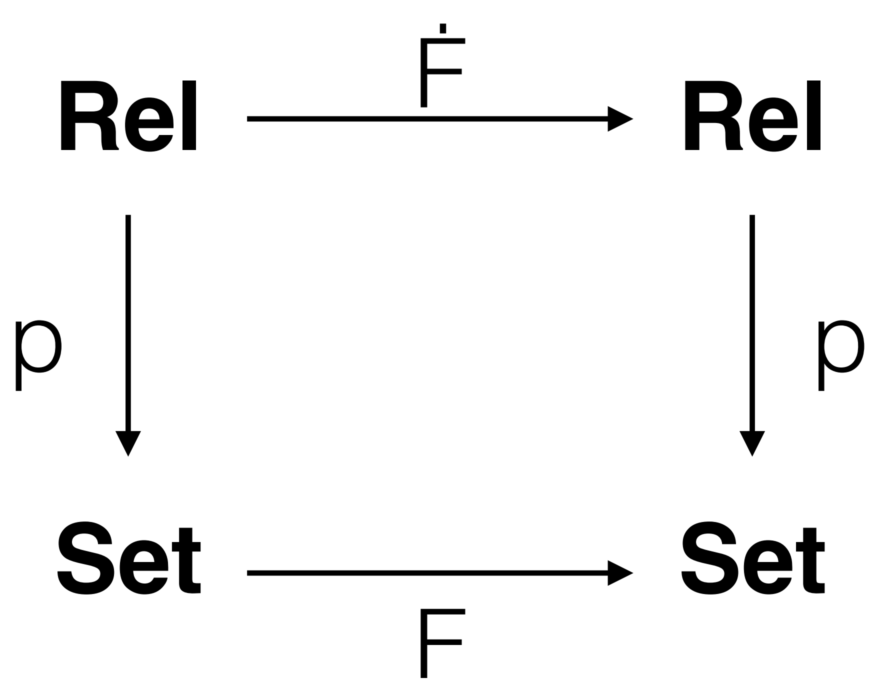
$$\begin{array}{ccc} \mathbf{Rel} & \xrightarrow{\dot{F}} & \mathbf{Rel} \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle p} \\ \mathbf{Set} & \xrightarrow[F]{} & \mathbf{Set} \end{array}$$

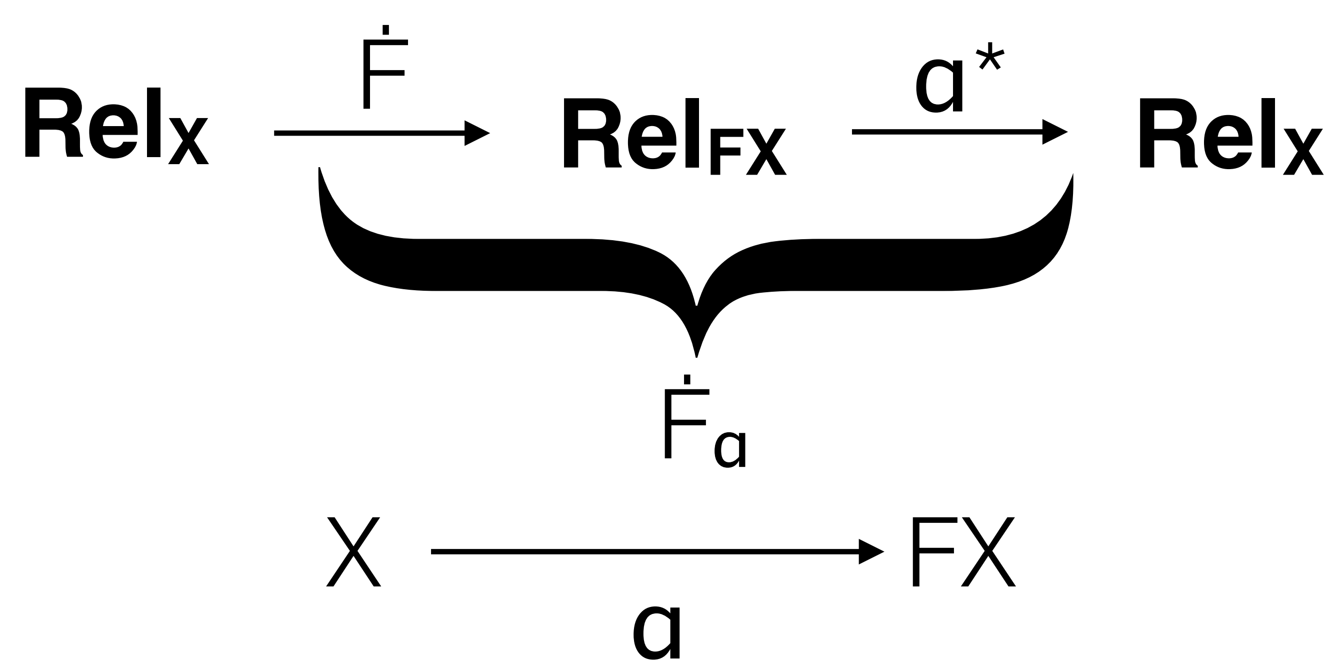
$$\mathbf{Rel}_X \xrightarrow{\dot{F}} \mathbf{Rel}_{FX} \xrightarrow{\alpha^*} \mathbf{Rel}_X$$

$$X \xrightarrow[\alpha]{} FX$$

Coinductive Predicates

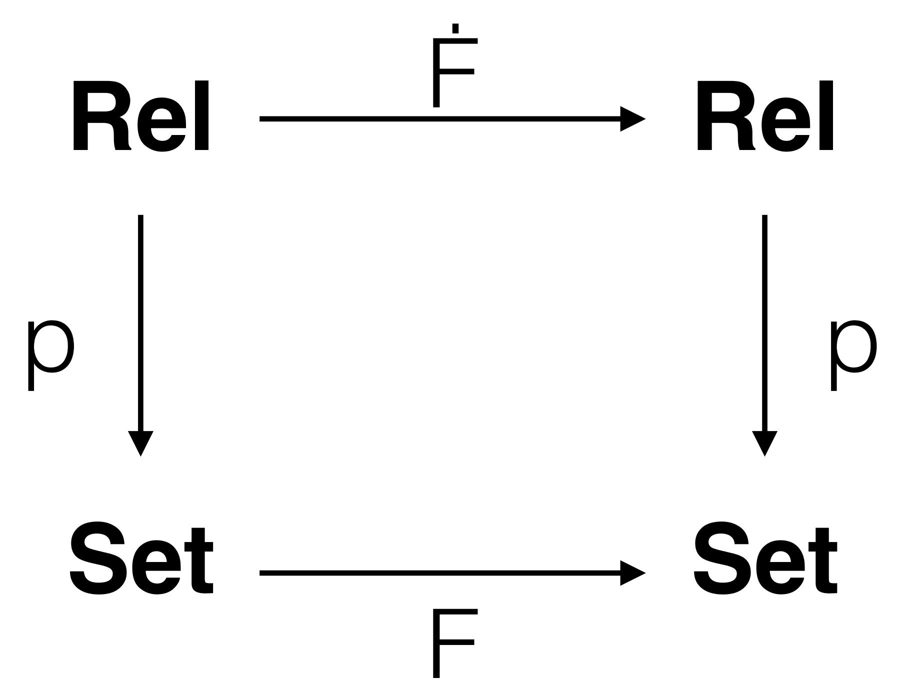
Hermida and Jacobs - Information and Computation 1998

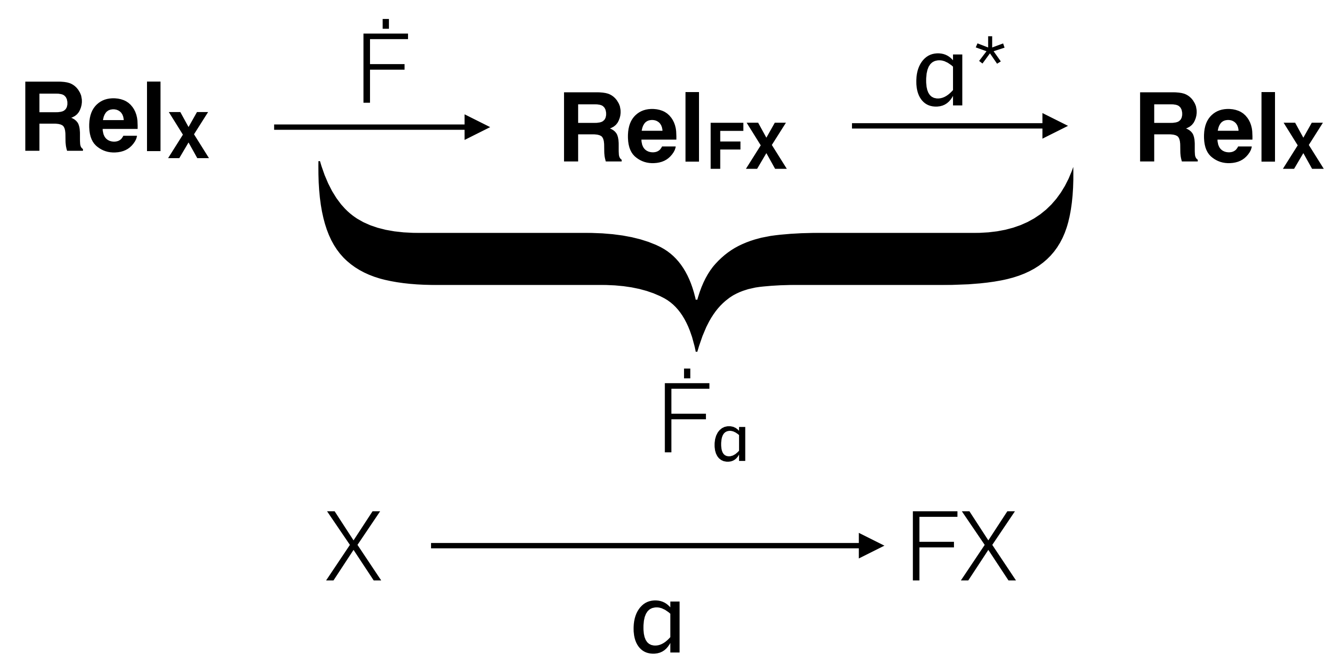




Coinductive Predicates

Hermida and Jacobs - Information and Computation 1998

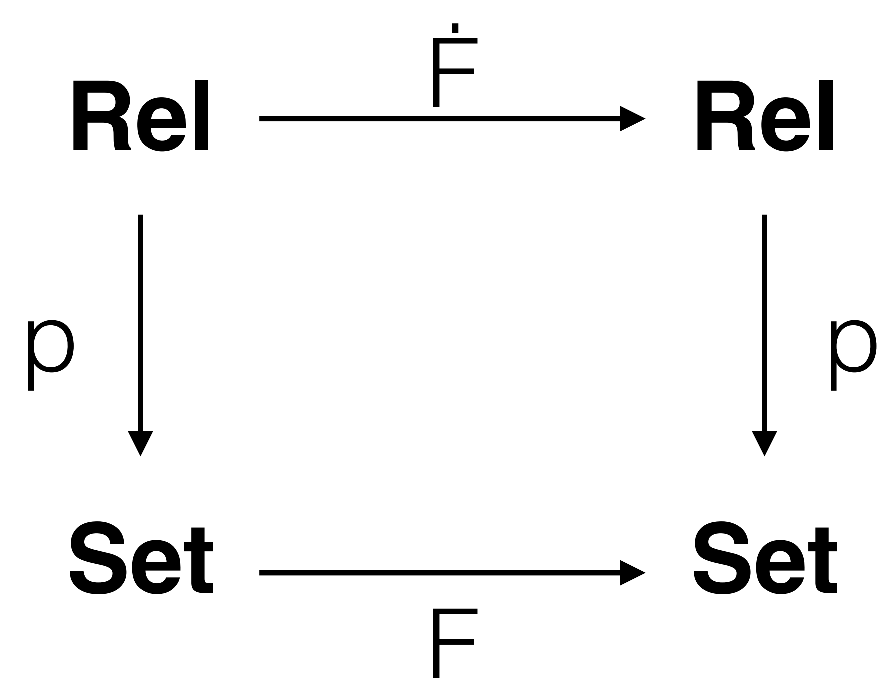


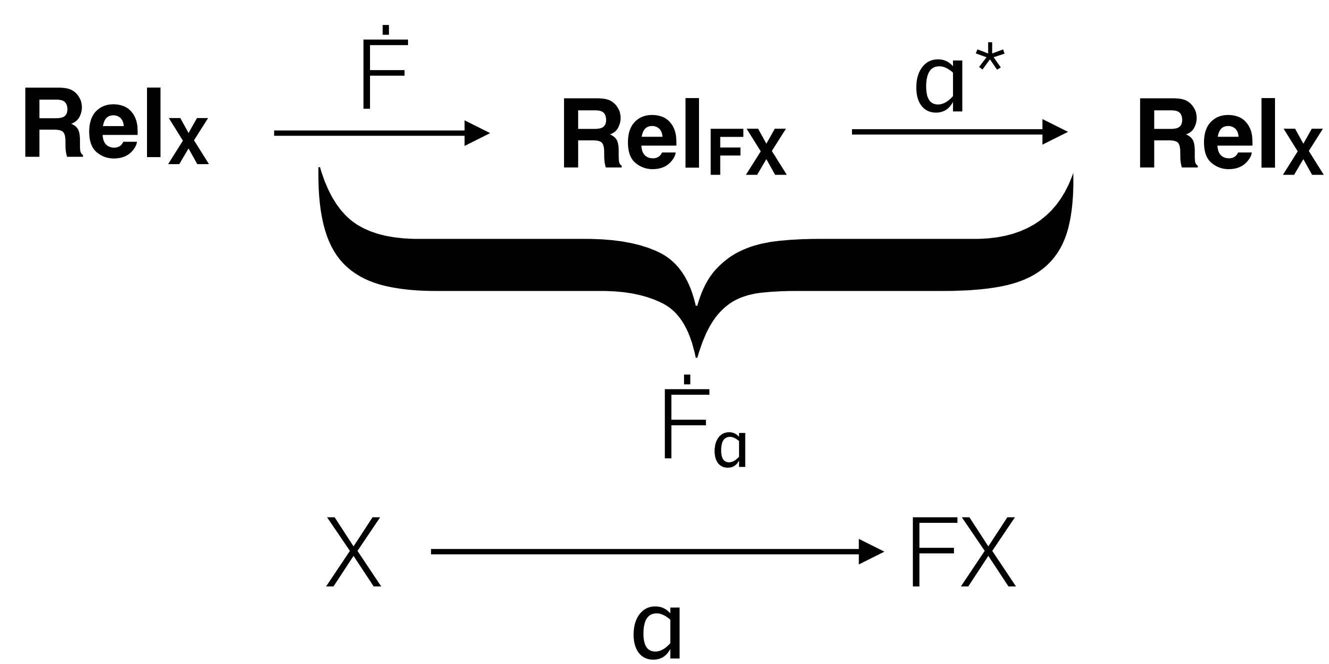


Coinductive Predicates

Hermida and Jacobs - Information and Computation 1998

$F = 2 \times Id^A$



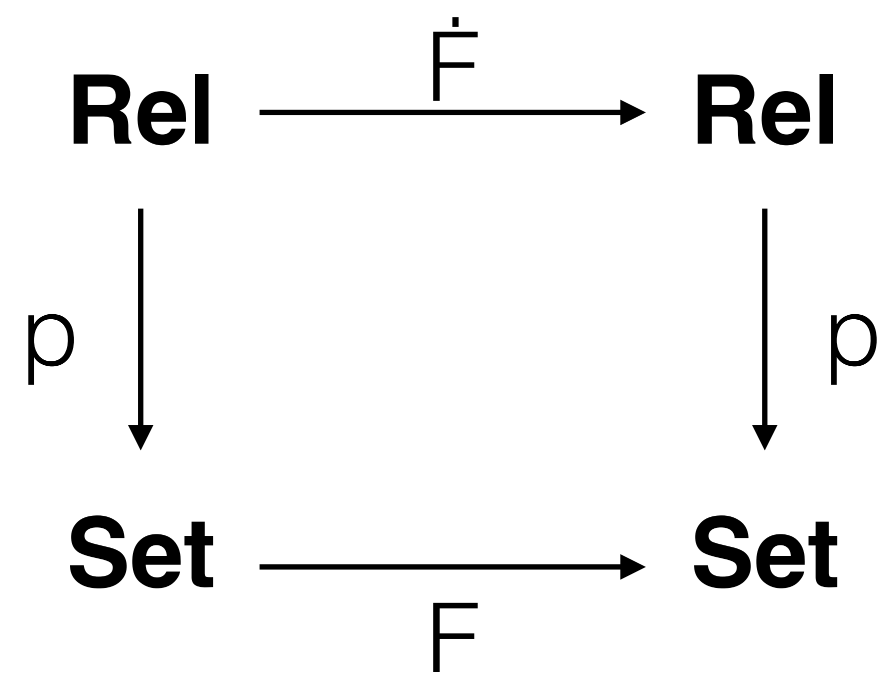


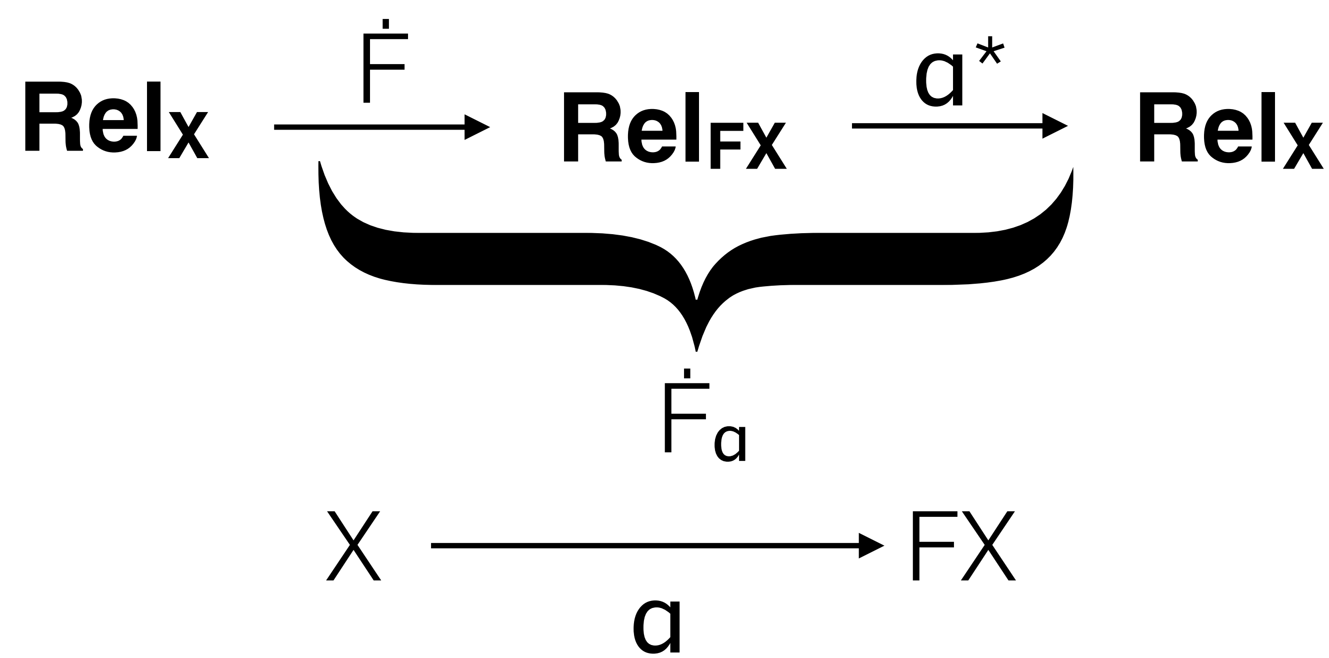
Coinductive Predicates

Hermida and Jacobs - Information and Computation 1998

$F = 2 \times Id^A$

$\dot{F}(R) = \{ ((p,f), (q,g)) \mid p = q \text{ and for all } a \in A \; f(a) \; R \; g(a) \}$





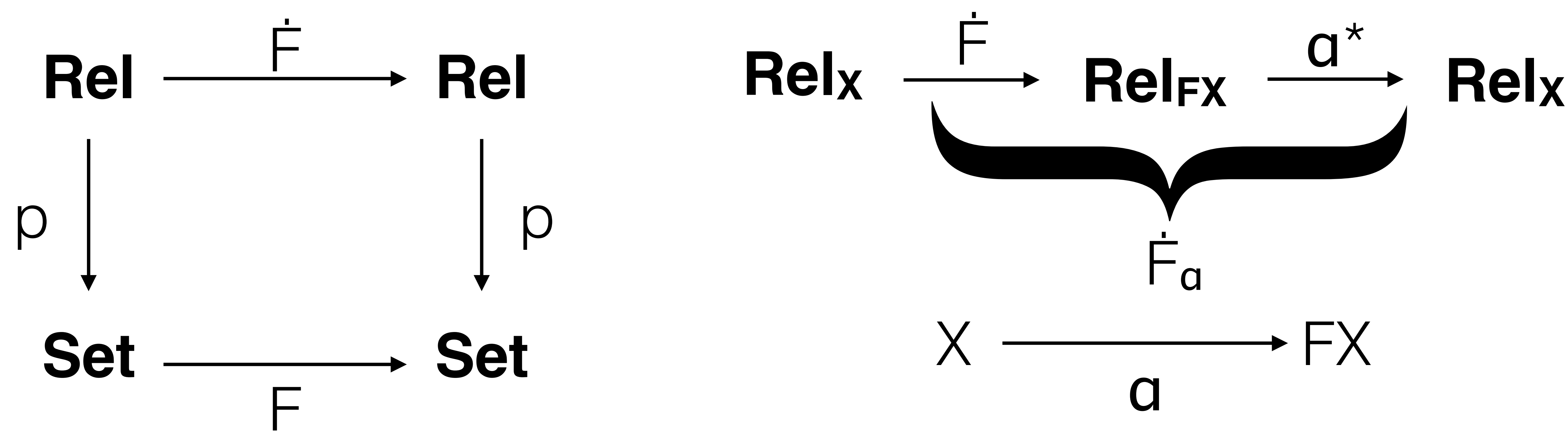
Coinductive Predicates

Hermida and Jacobs - Information and Computation 1998

$F = 2 \times Id^A$

$\dot{F}(R) = \{ ((p,f), (q,g)) \mid p = q \text{ and for all } a \in A\ f(a)\ R\ g(a) \}$

$\langle o,t \rangle : X \to 2 \times X^A$



Coinductive Predicates

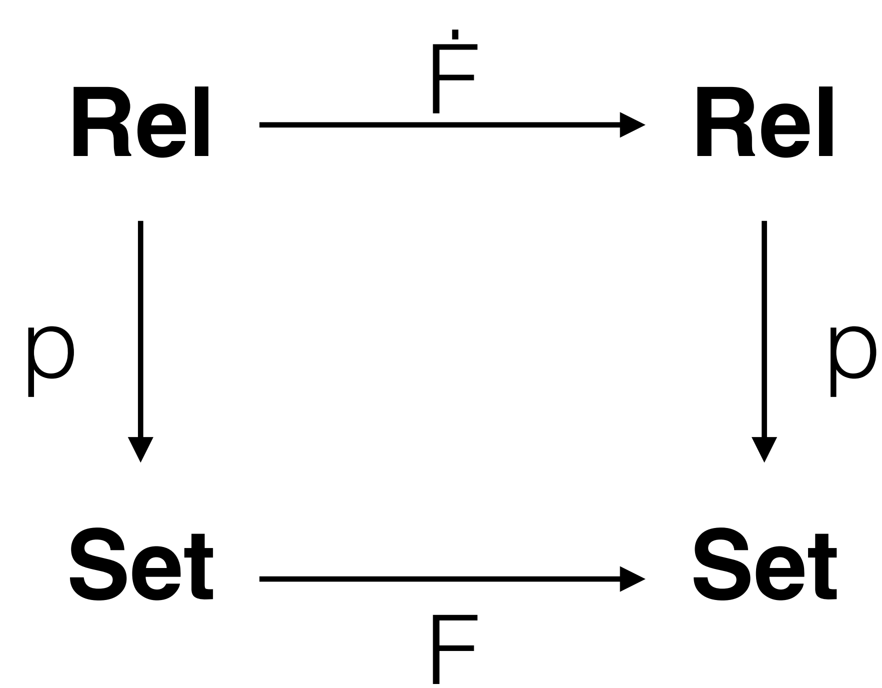
Hermida and Jacobs - Information and Computation 1998

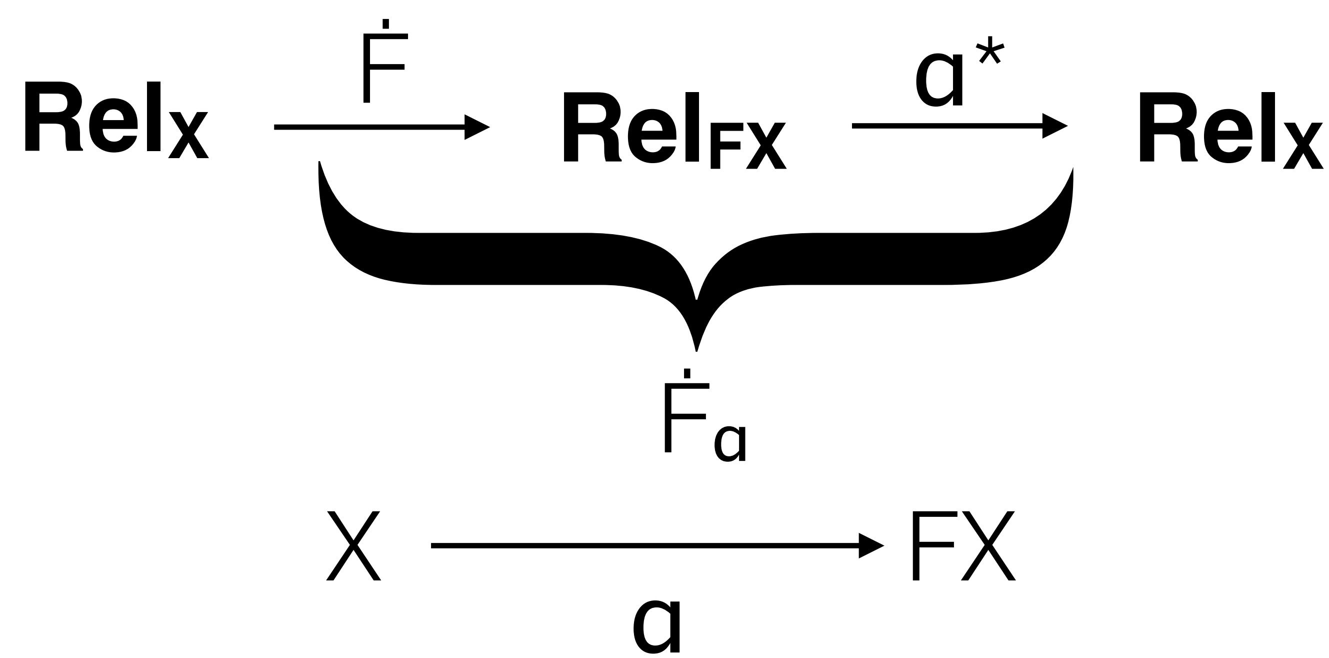
$$F=2\times Id^A$$

$$\dot{F}(R)=\{\ ((p,f),\ (q,g))\ |\ p=q \text{ and for all } a\in A\ f(a)\ R\ g(a)\}$$

$$\langle o,t\rangle : X \rightarrow 2\times X^A$$

$$\dot{F}_{\langle o,t\rangle}(R)=B(R)$$





Coinductive Predicates

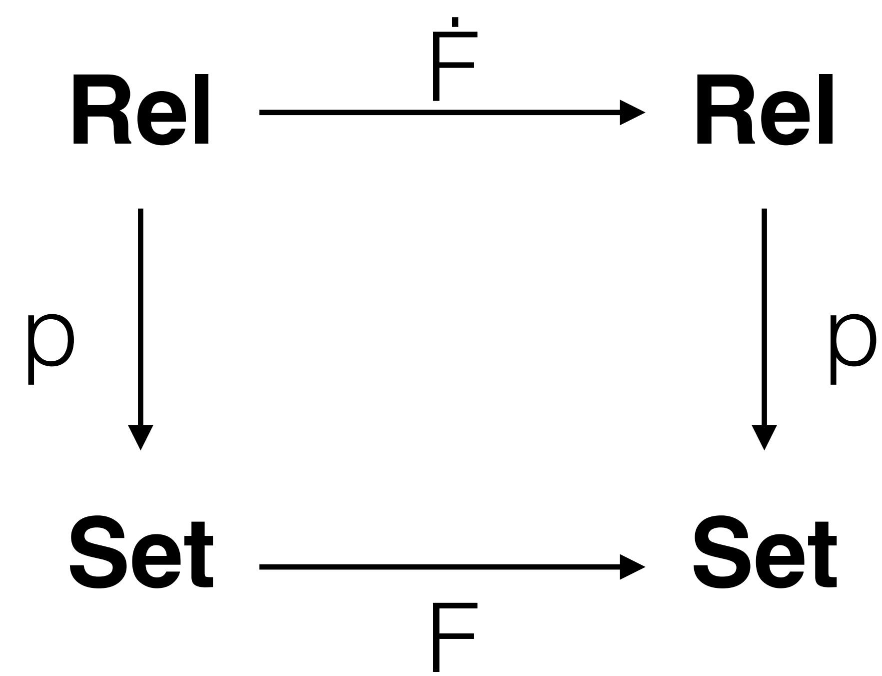
Hermida and Jacobs - Information and Computation 1998

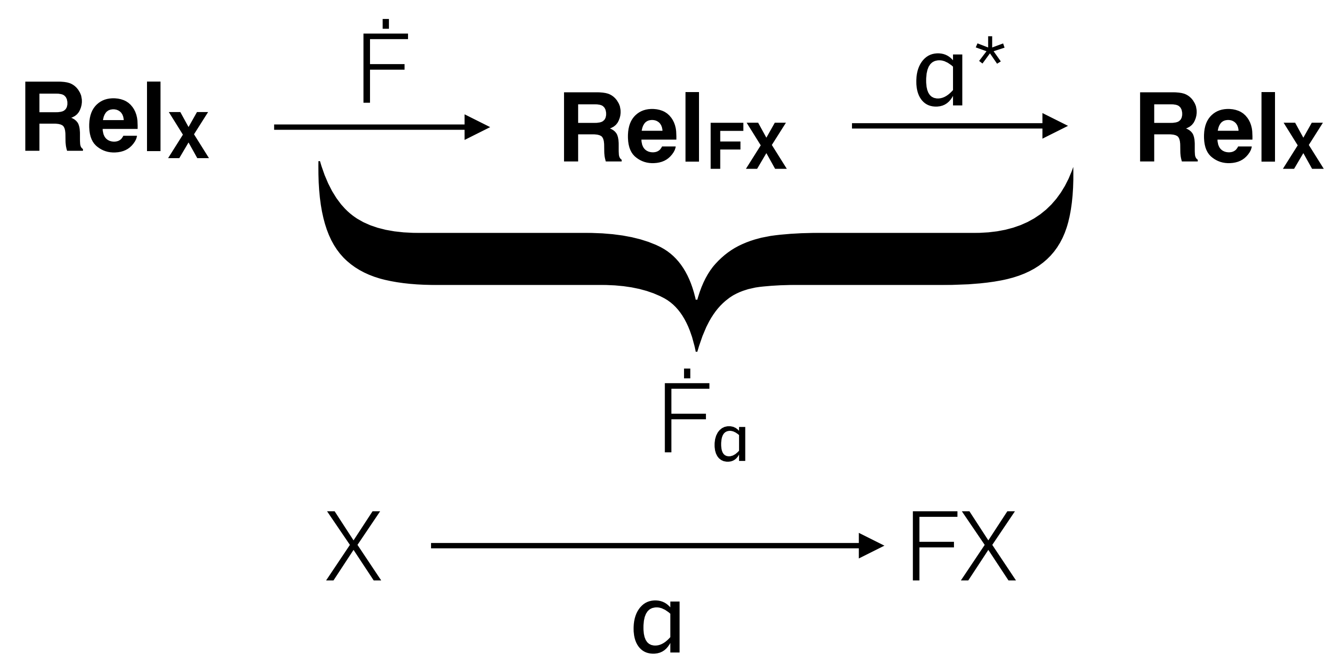
$F = 2 \times Id^A$

$\dot{F}(R) = \{ ((p,f), (q,g)) \mid p = q \text{ and for all } a \in A\ f(a)\ R\ g(a) \}$

$\langle o,t \rangle : X \to 2 \times X^A$

$\dot{F}_{\langle o,t \rangle}(R) = B(R)$





Actually, this can be defined for any bifibration

Coinductive Predicates

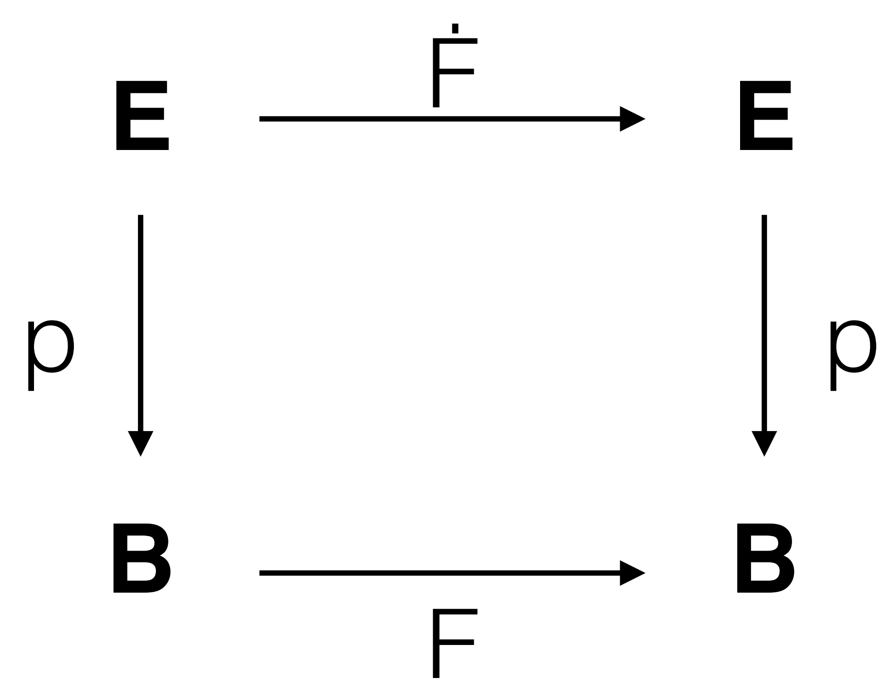
Hermida and Jacobs - Information and Computation 1998

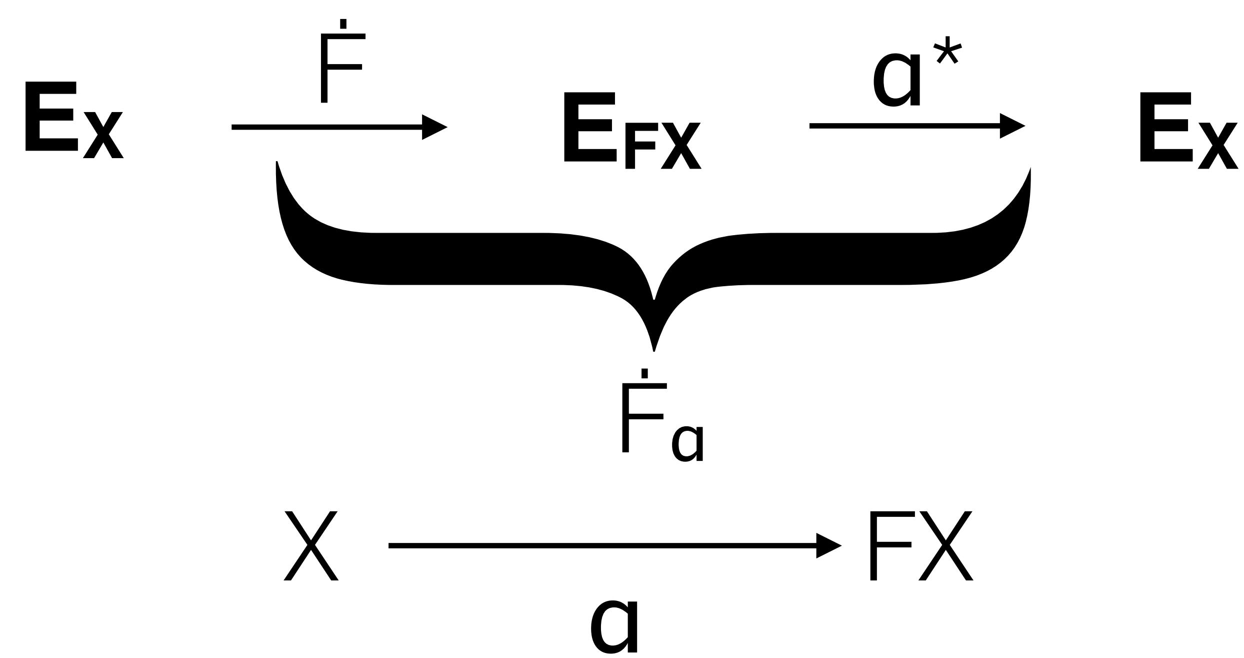
$$F = 2 \times Id^A$$

$$\dot{F}(R) = \{ ((p,f),(q,g)) \mid p = q \text{ and for all } a \in A \; f(a) \; R \; g(a) \}$$

$$\langle o,t \rangle : X \to 2 \times X^A$$

$$\dot{F}_{\langle o,t \rangle}(R) = B(R)$$





Actually, this can be defined for any bifibration

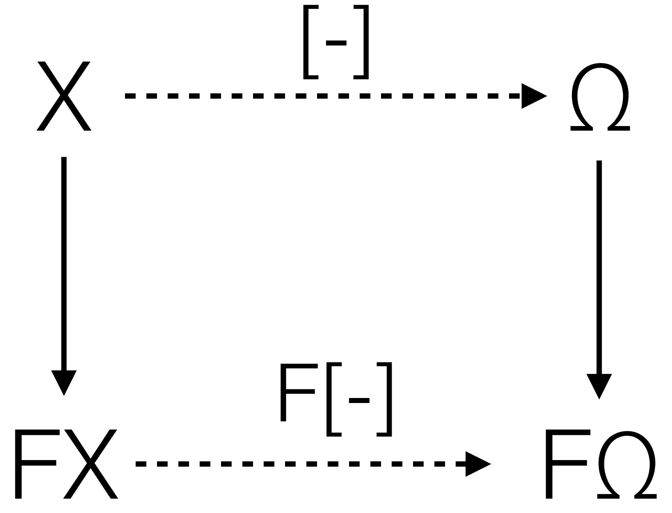
Compatibility of Bhv

Bhv: $\mathbf{Rel_X} \dashrightarrow \mathbf{Rel_X}$

$Bhv(R) = \{ (x,y) \mid x \sim x' \; R \; y' \sim y \}$

Compatibility of Bhv

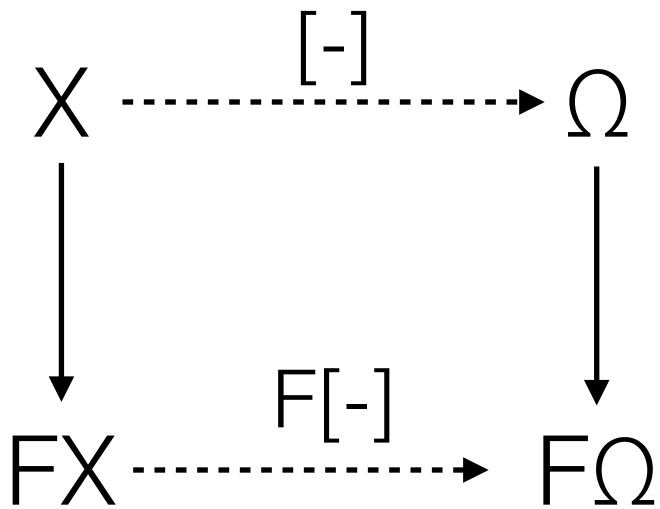
Bhv: **Rel$_X$**-->**Rel$_X$**
Bhv(R)= { (x,y) | x∼x' R y'∼y }



Compatibility of Bhv

Bhv: $\mathbf{Rel}_X \dashrightarrow \mathbf{Rel}_X$

$\mathrm{Bhv}(R) = \{\, (x,y) \mid x \sim x' \; R \; y' \sim y \,\}$

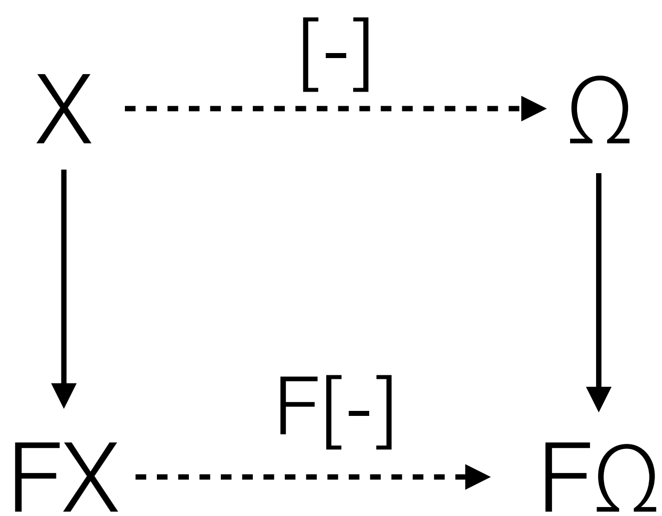


$x \sim y \overset{\text{def}}{\iff} [x]=[y]$

Compatibility of Bhv

Bhv: $\mathbf{Rel}_X \dashrightarrow \mathbf{Rel}_X$

$\mathrm{Bhv}(R) = \{ (x,y) \mid x \sim x' \; R \; y' \sim y \}$



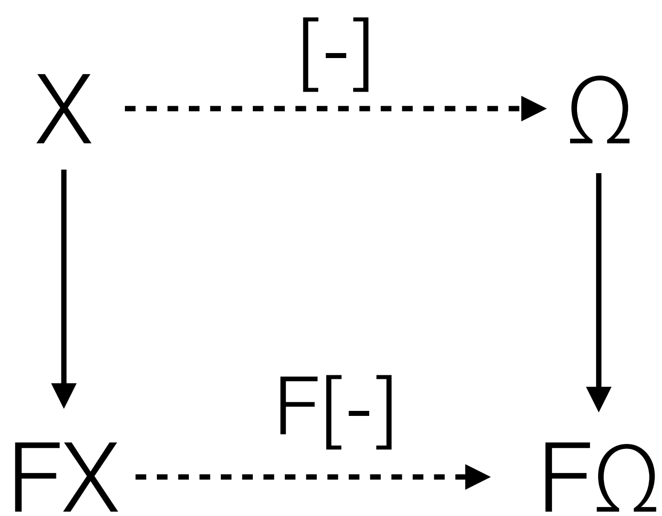
$$\mathbf{Rel}_X \xrightarrow{\coprod_{[-]}} \mathbf{Rel}_\Omega \xrightarrow{[-]^*} \mathbf{Rel}_X$$

$$x \sim y \overset{\text{def}}{\iff} [x] = [y]$$

Compatibility of Bhv

Bhv: **Rel$_X$**-->**Rel$_X$**

Bhv(R)= { (x,y) | x∼x' R y'∼y }



$$\mathbf{Rel}_X \xrightarrow{\coprod_{[-]}} \mathbf{Rel}_\Omega \xrightarrow{[-]^*} \mathbf{Rel}_X$$

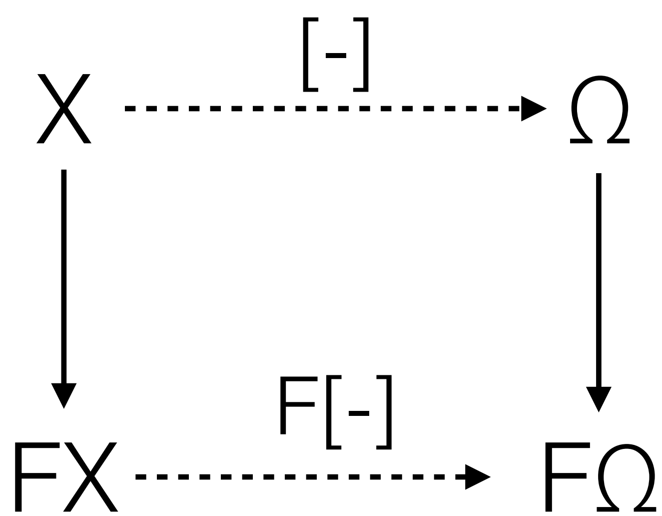
$$x \sim y \overset{\text{def}}{\Longleftrightarrow} [x]=[y]$$

$$\coprod_{[-]}(R)=\{ ([x'],[y']) \mid x' \; R \; y' \}$$

Compatibility of Bhv

Bhv: $\mathbf{Rel}_X \dashrightarrow \mathbf{Rel}_X$

$\mathrm{Bhv}(R) = \{ (x,y) \mid x \sim x' \; R \; y' \sim y \}$



$$\mathbf{Rel}_X \xrightarrow{\coprod_{[-]}} \mathbf{Rel}_\Omega \xrightarrow{[-]^*} \mathbf{Rel}_X$$

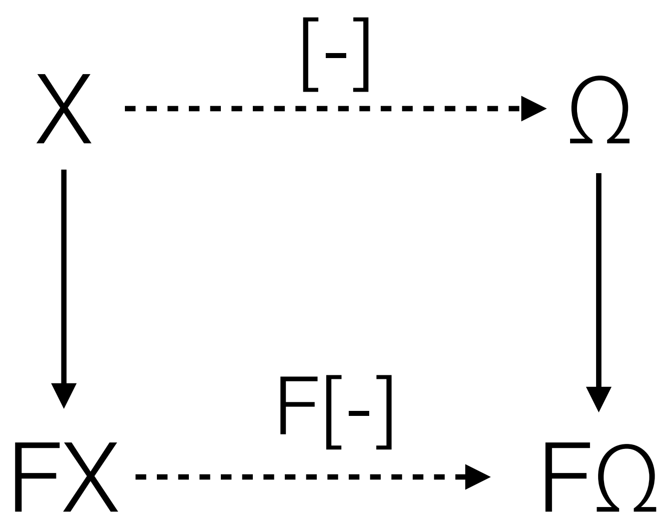
$$x \sim y \overset{\text{def}}{\Longleftrightarrow} [x]=[y]$$

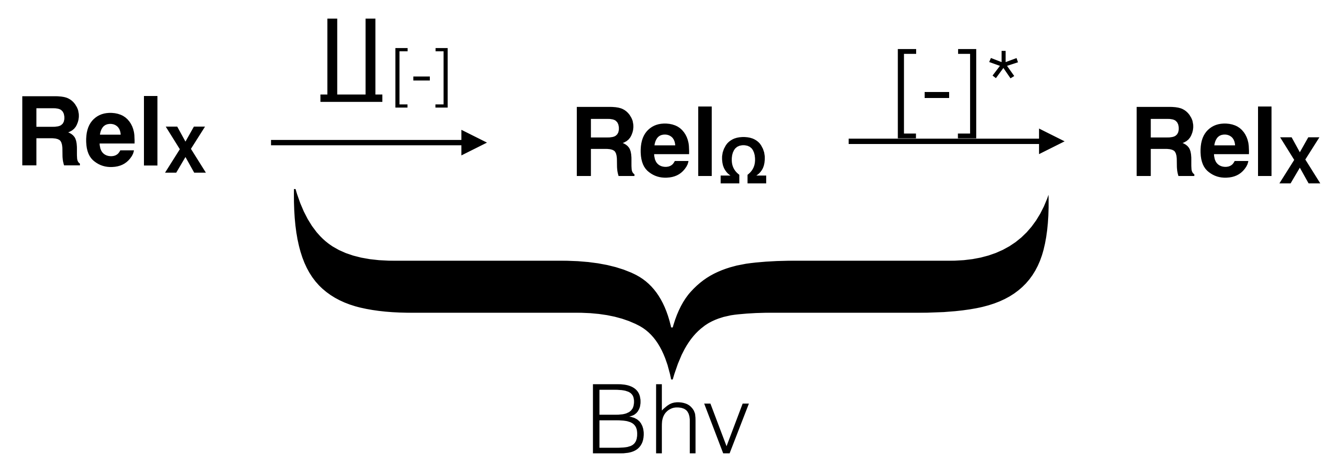
$$\coprod_{[-]}(R) = \{ ([x'],[y']) \mid x' \; R \; y' \}$$

$$[-]^* \coprod_{[-]}(R) = \{ (x,y) \mid [x]=[x'] \; R \; [y']=[y] \}$$

Compatibility of Bhv

$$\mathrm{Bhv}\colon \mathbf{Rel}_X \dashrightarrow \mathbf{Rel}_X$$
$$\mathrm{Bhv}(R) = \{\, (x,y) \mid x \sim x' \; R \; y' \sim y \,\}$$





$$x \sim y \overset{\text{def}}{\iff} [x]=[y]$$

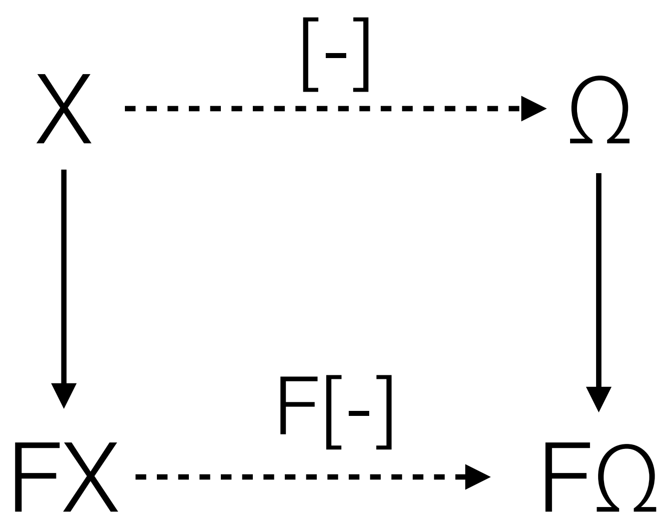
$$\coprod_{[-]}(R) = \{\, ([x'],[y']) \mid x' \; R \; y' \,\}$$

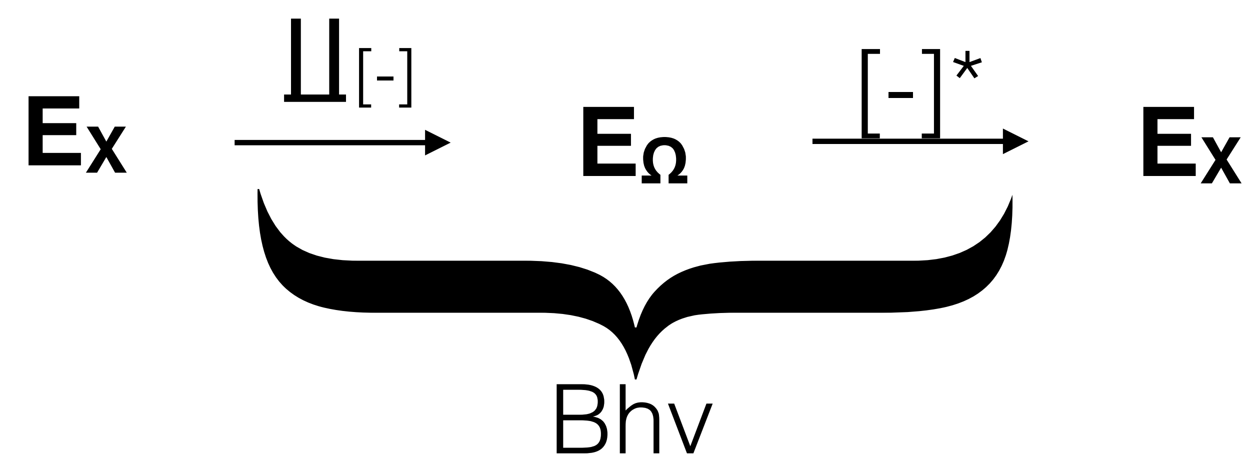
$$[-]^* \coprod_{[-]}(R) = \{\, (x,y) \mid [x]=[x'] \; R \; [y']=[y] \,\}$$

Compatibility of Bhv

Bhv: $\mathbf{Rel}_X \dashrightarrow \mathbf{Rel}_X$

$Bhv(R)= \{ (x,y) \mid x \sim x' \; R \; y' \sim y \}$





$x \sim y \overset{def}{\iff} [x]=[y]$

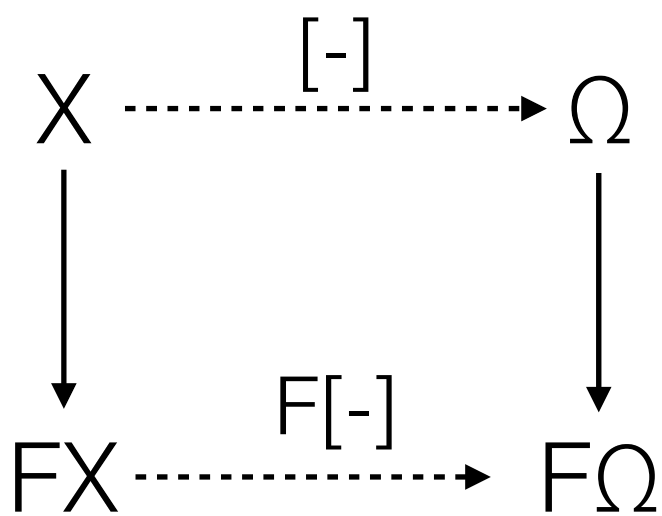
$\coprod_{[-]}(R)=\{ ([x'],[y']) \mid x' \; R \; y' \}$

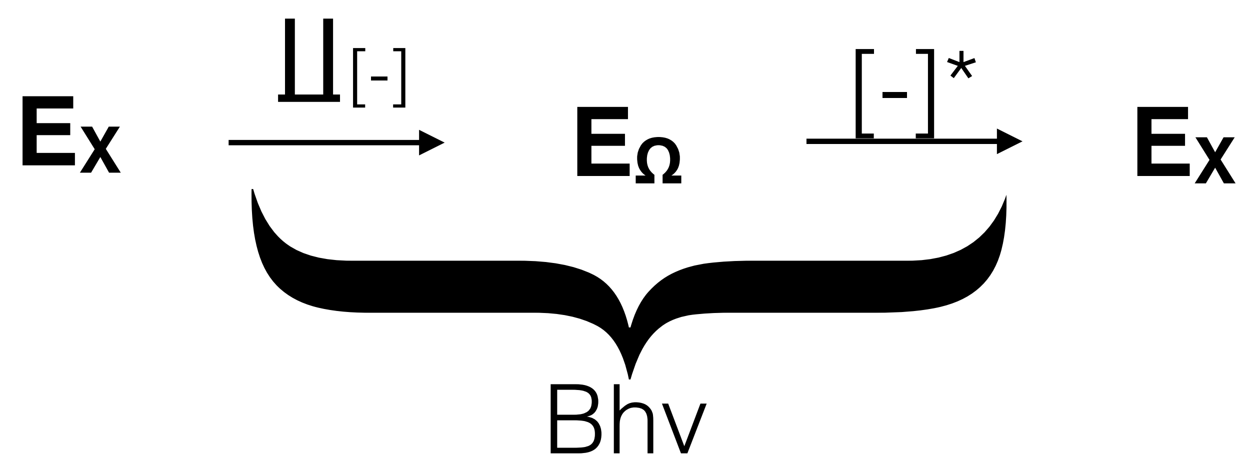
$[-]^*\coprod_{[-]}(R)=\{ (x,y) \mid [x]=[x'] \; R \; [y']=[y] \}$

Compatibility of Bhv

Bhv: $\mathbf{Rel}_X \dashrightarrow \mathbf{Rel}_X$

$\mathrm{Bhv}(R) = \{\, (x,y) \mid x \sim x' \; R \; y' \sim y \,\}$



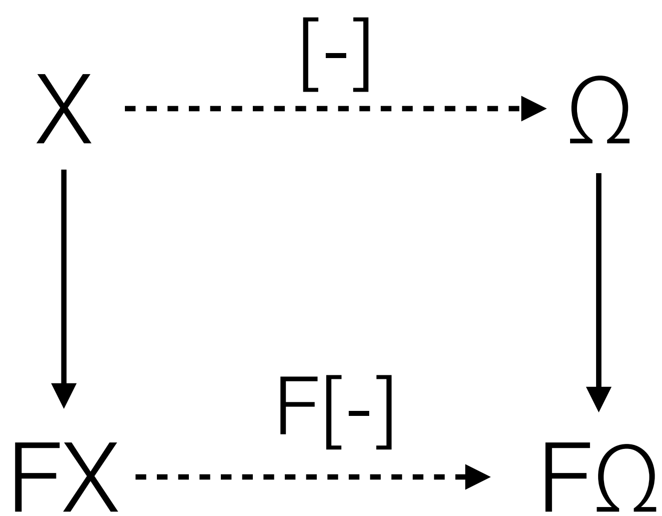


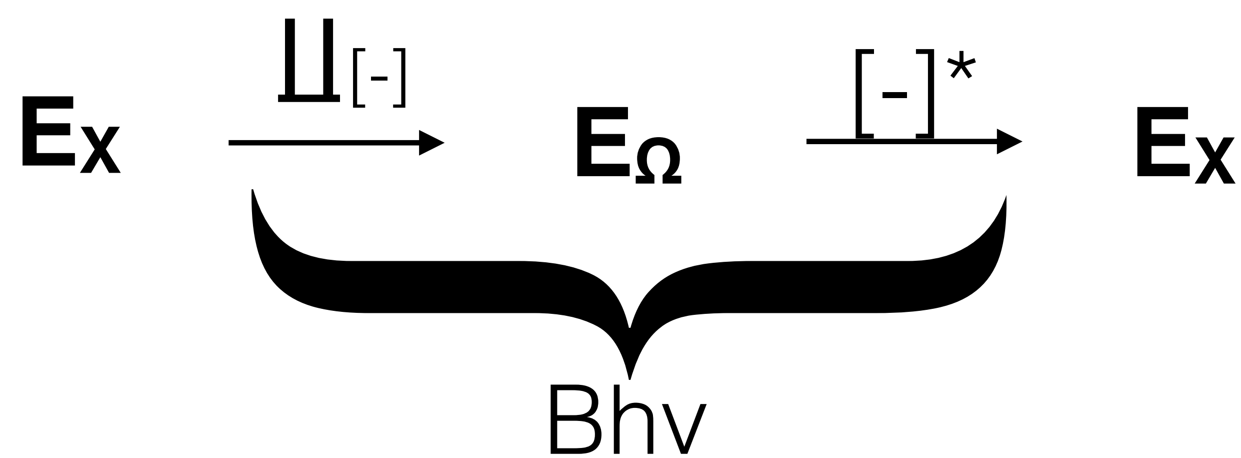
$x \sim y \overset{\text{def}}{\iff} [x]=[y]$

Compatibility of Bhv

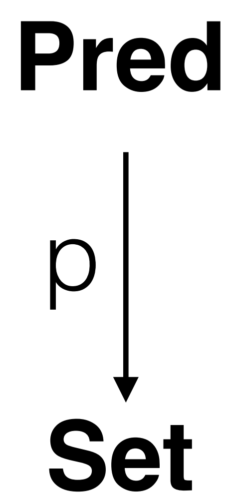
Bhv: $\mathbf{Rel}_X \dashrightarrow \mathbf{Rel}_X$

$Bhv(R) = \{ (x,y) \mid x \sim x' \; R \; y' \sim y \}$





$x \sim y \overset{def}{\Longleftrightarrow} [x]=[y]$



$Bhv(P \subseteq X) = \{x \mid x \sim x' \in P\}$

Compatibility of Bhv

Theorem: Let $(\dot{F}, F)$ be a *fibration map* and $\alpha : X \to FX$ be an F-coalgebra then Beh is compatible with $\dot{F}_{\alpha}$

Compatibility of Bhv

Theorem: Let $(\dot{F}, F)$ be a *fibration map* and $\boldsymbol{\alpha}: X \to FX$ be an F-coalgebra then Beh is compatible with $\dot{F}_{\boldsymbol{\alpha}}$

Corollary:
For the monotone predicate lifting (in Coalgebraic modal logic) up-to Beh is compatible

Compatibility of Bhv

<u>Theorem</u>: Let ($\dot{F}$,F) be a *fibration map* and **α**:X→FX be an F-coalgebra then Beh is compatible with $\dot{F}_{\alpha}$

Compatibility of Bhv

Theorem: Let $(\dot{F},F)$ be a *fibration map* and $\boldsymbol{\alpha}:X\to FX$ be an F-coalgebra then Beh is compatible with $\dot{F}_{\boldsymbol{\alpha}}$

Whenever F preserves weak pullbacks
the *canonical relational lifting* is a fibration map

Compatibility of Bhv

Theorem: Let $(\dot{F},F)$ be a *fibration map* and $\alpha: X \to FX$ be an F-coalgebra then Beh is compatible with $\dot{F}_\alpha$

Whenever F preserves weak pullbacks
the *canonical relational lifting* is a fibration map

Corollary:
up-to language equivalence (at the beginning of this talk) and up-to bisimilarity (Milner) are compatible

References

- Bonchi, Petrisan, Pous, Rot: Coinduction up-to in a fibrational setting. LICS 2014
- Bonchi, Petrisan, Pous, Rot: Lax bialgebra and up-to technique for weak bisimulation. CONCUR 2015
- Bonchi, Petrisan, Pous, Rot: A general account of bisimulation up-to. Submitted to ACTA
- Rot: Enhanced coinduction. Ph.D. Thesis, Leiden Univ.