Principle

In a 1933 lecture and paper titled *On the Method of Theoretical Physics*, Albert Einstein stated:

> *"It can scarcely be denied that the supreme goal of all theory is to make the irreducible basic elements as simple and as few as possible without having to surrender the adequate representation of a single datum of experience."*

Albert Einstein did not actually say the exact famous phrase:

"*everything should be made as simple as possible, but not simpler,*“

nor

"*everything should be made as abstract as possible, but not abstracter,*"

Fair Merge

Time after Time

Manfred Broy

Technische Universität München
Institut für Informatik
D-80290 Munich, Germany

On the passage of time in behaviors and their abstractions

- Most behavior models in computing theories are time agnostic – untimed

- Some applications such as in real time programming require time (more precisely, require models of behavior with explicit time):
  - ◊ Example: Interrupts, reactions within time bounds, time outs, cyber-physical systems, …

- There are behaviors which can be described in an untimed way, but arguing about their behaviors in certain contexts timed considerations seem unavoidable

Merging finite sequences

$$\text{Merge: Data}^* \times \text{Data}^* \to \wp(\text{Data}^*)$$

$$\text{Merge}(x, \langle\rangle) = \text{Merge}(\langle\rangle, x) = \{x\}$$

$$\text{Merge}(\langle a\rangle\hat{\ }x, \langle b\rangle\hat{\ }z) =$$
$$\{\langle a\rangle\hat{\ }y: y \in \text{Merge}(x, \langle b\rangle\hat{\ }z)\} \cup \{\langle b\rangle\hat{\ }y: y \in \text{Merge}(\langle a\rangle\hat{\ }x, z)\}$$

Here $x\hat{\ }z$ denotes the concatenation of x and z!

Fair Merge

What a fair merge of two possibly infinite data streams is,
may intuitively be quite clear.

Given two finite or infinite data streams x and z, the set of fair merges consists of all streams which consist of completely merging the two sub-streams x and z.

However, a formal definition of fair merge is a bit tricky!

Task: given two data streams x and z define the set of their fair merges!

A first try: A straightforward (?) naïve recursive definition

$$\text{FairMerge}: D^{*|\omega} \times D^{*|\omega} \to \wp(D^{*|\omega})$$

$$\text{FairMerge}(x, \langle\rangle) = \text{FairMerge}(\langle\rangle, x) = \{x\}$$

$$\text{FairMerge}(\langle a\rangle\hat{\ }x, \langle b\rangle\hat{\ }z) =$$
$$\{\langle a\rangle\hat{\ }y : y \in \text{FairMerge}(x, \langle b\rangle\hat{\ }z)\} \cup \{\langle b\rangle\hat{\ }y : y \in \text{FairMerge}(\langle a\rangle\hat{\ }x, z)\}$$

Problem: If $b \neq d$ then

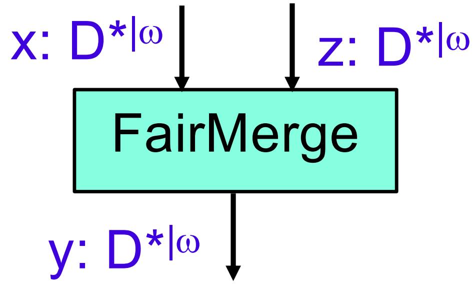
$$\neg(d^\infty \in \text{FairMerge}(d^\infty, \langle b\rangle))$$

cannot be proved!

This definition is insufficient!

Fair Merge

How do we specify a fair merge of two data streams x and z?



A not very elegant brute force specification:

$$\text{FairMerge}(x, z) = \{y \in D^{*|\omega}: \exists\, c \in \{1, 2\}^{*|\omega}: 1\#c = \#x \wedge 2\#c = \#y \wedge \text{split}(1, c, y) = x \wedge \text{split}(2, c, y) = z\}$$

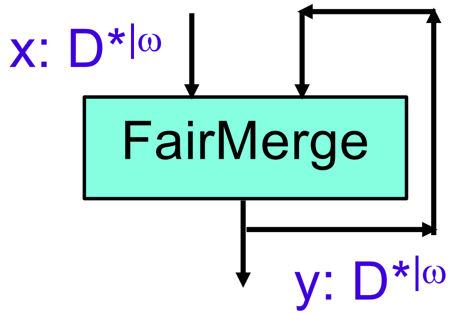
where c is the distributer:

$\text{split}(1, \langle 1\rangle\hat{}c, \langle m\rangle\hat{}y) = \langle m\rangle\hat{}\text{split}(1, c, y)$ $\quad$ $\text{split}(2, \langle 2\rangle\hat{}c, \langle m\rangle\hat{}y) = \langle m\rangle\hat{}\text{split}(2, c, y)$

$\text{split}(2, \langle 1\rangle\hat{}c, \langle m\rangle\hat{}y) = \text{split}(2, c, y)$ $\quad$ $\text{split}(1, \langle 2\rangle\hat{}c, \langle m\rangle\hat{}y) = \text{split}(1, c, y)$

$\text{split}(n, c, \langle\rangle) = \langle\rangle$

Feedback for FairMerge



$$y \in \text{FairMerge}(x, y)$$

Example: Using our definition:

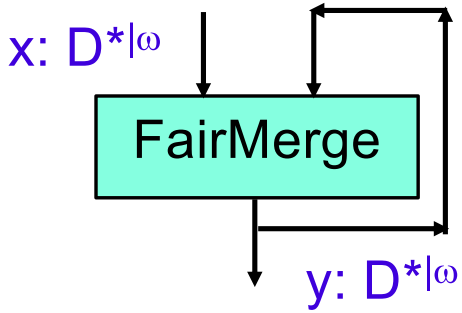
$$y \in \text{FairMerge}(\langle 1 \rangle, y)$$

$$\Leftrightarrow$$

$$\exists\ c \in \{1, 2\}^{*|\omega}: 1\#c = 1 \wedge 2\#c = \#y \wedge \text{split}(1, c, y) = \langle 1 \rangle \wedge \text{split}(2, c, y) = y$$

Is this sufficiently describing feedback?

Feedback for Fair Merge



$$y \in \text{FairMerge}(\langle 1 \rangle, y)$$

$$(2\hat{\ }1)^{\omega} \in \text{FairMerge}(\langle 1 \rangle, (2\hat{\ }1)^{\omega})$$

?

Untimed Streams

$$D^{*|\omega} = D^{*} \cup D^{\omega}$$

Finite Streams: $D^{*} = \cup_{n \in \mathbb{N}} [1:n] \rightarrow D$

Infinite Streams: $D^{\omega} = \mathbb{N}_{+} \rightarrow D$

$$[1:n] = \{t \in \mathbb{N} : 1 \leq t \leq n\}$$

Timed Streams

Timed streams $(D^*)^\omega = \mathbb{N}_+ \to D^*$

Finite timed streams $(D^*)^* = \cup_{n \in \mathbb{N}} ([1:n] \to D^*)$

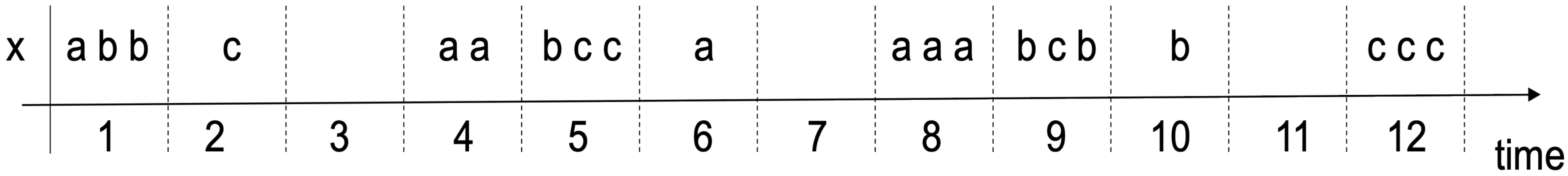
$$x{\downarrow}t : \{n \in \mathbb{N}: 1 \le n \le t\} \to D^*$$
$$1 \le n \le t \Rightarrow (x{\downarrow}t)(n) = x(n)$$
$$S \subseteq (D^*)^\omega: S{\downarrow}t = \{x{\downarrow}t: x \in S\}$$

$\#x$ number of elements in stream x

$S\#x$ number of elements in stream x that are in set S

$$d\#x = \{d\}\#x$$

Timed Streams: Illustration - Time Flow and Data Flow



Timed stream $x = \langle\langle a\ b\ b\rangle\ \langle c\rangle\ \langle\rangle\ \langle a\ a\rangle\ \langle b\ c\ c\rangle\ \langle a\rangle\ \langle\rangle\ \langle a\ a\ a\rangle\ \langle b\ c\ b\rangle\ \langle b\rangle\ \langle\rangle\ \langle c\ c\ c\rangle\ \dots\ \rangle$

Time abstraction $\bar{x} = \langle a\ b\ b\ c\ a\ a\ b\ c\ c\ a\ a\ a\ b\ c\ b\ b\ c\ c\ c\ \dots\rangle$

Partial orders

Define a partial order for nondeterministic/set valued functions on streams

prefix ordering on streams: $x \sqsubseteq y = \exists\, z: x\hat{\ }z = y$

Egli-Milner preorder on sets of streams (anti-symmetry does not hold):

$$R \sqsubseteq S = (\forall\, r \in R : \exists\, s \in S: r \sqsubseteq s \wedge \forall\, s \in S : \exists\, r \in R: r \sqsubseteq s)$$

Preordering on nondeterministic functions on streams:

$$F, G: D^{*|\omega} \rightarrow \wp(D^{*|\omega})\backslash\{\varnothing\}$$

$$F \sqsubseteq G = \forall\ s \in D^{*|\omega}: F(s) \sqsubseteq G(s)$$

Untimed streams versus Timed streams

Let $x, y \in D^{*|\omega}$

Prefix ordering: $x \sqsubseteq y = \exists z: x\hat{\ }z = y$

Untimed stream function:

$$f: D^{*|\omega} \to D^{*|\omega}$$

$$F: D^{*|\omega} \to \wp(D^{*|\omega})\backslash\{\emptyset\}$$

f is monotonic if

$$x \sqsubseteq z \Rightarrow f(x) \sqsubseteq f(z)$$

F is monotonic if

$$x \sqsubseteq z \Rightarrow F(x) \sqsubseteq F(z)$$

Let $x, y \in (D^*)^\omega$, $t \in \mathbb{N}$

Timed stream function:

$$f: (D^*)^\omega \to (D^*)^\omega$$

$$F: (D^*)^\omega \to \wp((D^*)^\omega)\backslash\{\emptyset\}$$

Strong causality:

$$x\downarrow t = z\downarrow t \Rightarrow f(x)\downarrow t+1 = f(z)\downarrow t+1$$

$$x\downarrow t = z\downarrow t \Rightarrow F(x)\downarrow t+1 = F(z)\downarrow t+1$$

We write

SC[f]

to express that f is strongly causal

Fully Realizable Timed Behaviors

Given $F: (D^*)^\omega \to \wp((D^*)^\omega) \setminus \{\emptyset\}$

$$Real[F] = \{f \in (D^*)^\omega \to (D^*)^\omega: SC[f] \wedge \forall\, x \in \vec{X}: f(x) \in F(x))\}$$

Real[F] denotes the set of realizations of F

F is realizable if $\exists\, f \in Real[F]$

F is fully realizable if F is realizable and

$$(y \in F(x)) = \exists\, f \in Real[F]: y = f(x)$$

Every realization $f \in Real[F]$ is a deterministic refinement, which

- defines a deterministic strategy to compute $y = f(x)$ such that $y \in F(x)$ holds
- has a unique fixpoint $x = f(x)$; then $x \in F(x)$

Time abstraction: relating time and untimed streams

For a timed stream $s \in (D^*)^\omega$ its time abstraction $\bar{s} \in D^{*|\omega}$ is defined by

$$\bar{s} = s(1)\hat{}s(2)\hat{}s(3\}\hat{}\ \ldots$$

For a timed function $f: (D^*)^\omega \rightarrow (D^*)^\omega$ its time abstraction

$$\bar{f} \in (D^*)^\omega \rightarrow \wp((D^*)^\omega)\backslash\{\varnothing\}$$

is defined by

$$\bar{f}(x') = \{\bar{y} : \exists\ x \in (D^*)^\omega : x' = \bar{x} \wedge y = f(x)\}$$

and for

$$F: (D^*)^\omega \rightarrow \wp((D^*)^\omega)\backslash\{\varnothing\}$$

by

$$\bar{F}(x') = \{\bar{y} : \exists\ x \in (D^*)^\omega : x' = \bar{x} \wedge y \in F(x)\}$$

Empty Streams

Untimed empty stream: $\langle\rangle$

Timed empty stream: ε where $\overline{\varepsilon} = \langle\rangle$

$$\varepsilon = \langle\langle\rangle\rangle\hat{}\,\varepsilon$$

Time insensitive timed behaviors

Timed functions

$$F: (D^*)^\omega \to \wp((D^*)^\omega)\backslash\{\emptyset\}$$

are called *time insensitive*, if for all $x, z \in (D^*)^\omega$

$$\overline{x} = \overline{z} \Rightarrow \overline{F(x)} = \overline{F(z)}$$

FairMerge is time insensitive!

Fair Merge for timed streams

Look at fair merge: We define functions on timed streams

$$\text{TFM}: (D^*)^\omega \times (D^*)^\omega \to \wp((D^*)^\omega)\backslash\{\emptyset\}$$
$$\text{TSM}: (D^*)^\omega \times (D^*)^\omega \to \wp((D^*)^\omega)\backslash\{\emptyset\}$$
$$\text{Timeshift}: (D^*)^\omega \to \wp((D^*)^\omega)\backslash\{\emptyset\}$$

such that fair merge TFM is defined by

$$\text{TFM}(x, z) = \{y \in (D^*)^\omega: y \in \text{TSM}(\text{Timeshift}(x), \text{Timeshift}(y))\}$$

where TSM defines the strongly causal fair merge without Timeshift

$$\text{TSM}(x, z) = \{y \in (D^*)^\omega: y(1) = \langle\rangle \wedge \forall\ t \in \mathbb{N}_+: y(t+1) \in \text{Merge}(x(t), z(t))\ \}$$

$$\text{Timeshift}(y) = \{s \in (D^*)^\omega : \bar{s} = \bar{y} \wedge \forall\ \ t \in \mathbb{N}: \overline{s{\downarrow}t} \sqsubseteq \overline{y{\downarrow}t}\ \}$$

Fair merge by abstraction of timed fair merge

The function TFM is time insensitive, however, $\overline{\text{TFM}}$ is not prefix monotonic in any of the classical power domain orderings.

$$\text{FairMerge} = \overline{\text{TFM}}$$

The function TFM is an instance for a class of strongly causal and fully realizable, time insensitive functions the time abstraction of which is not prefix monotonic.

$$y \in \text{TFM}(x, z) \Rightarrow \overline{y} \in \text{FairMerge}(\overline{x}, \overline{z})$$

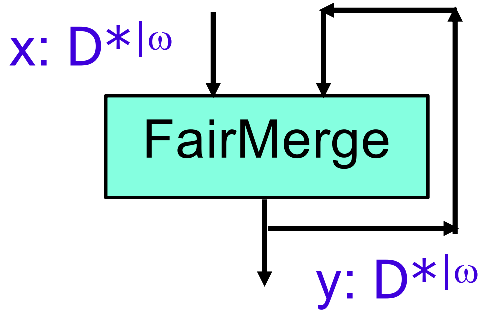
This shows that there exists a class of nondeterministic functions on untimed streams which are interactively computable by time abstraction but are not prefix monotonic.

We get for fixpoints y

$$y \in \text{TFM}(x, y) \Rightarrow \overline{y} \in \text{FairMerge}(\overline{x}, \overline{y})$$

Feedback for Fair Merge

$$\text{FairMerge} = \overline{\text{TFM}}$$



$$y \in \text{FairMerge}(x, y)$$

$$\text{FairMerge}(x', z') = \{y \in \overline{\text{TFM}(x, z)}: x' = \bar{x} \wedge z' = \bar{z}\}$$

$$\bar{x} = \langle 1 \rangle \wedge \; y \in \text{TFM}(\langle 1 \rangle, y) \Rightarrow \bar{y} = 1^{\infty}$$

$$y' \in \text{FairMerge}(\langle 1 \rangle, y') \cancel{\Rightarrow} y' = 1^{\infty}$$

$$(2\hat{\ }1)^{\omega} \in \text{FairMerge}(\langle 1 \rangle, (2\hat{\ }1)^{\omega})$$

Is FairMerge is prefix monotonic?

Consider properties the function FairMerge:

$$\text{FairMerge: } D^{*|\omega} \times D^{*|\omega} \rightarrow \wp(D^{*|\omega})$$

$$\text{FairMerge}(1^\infty, \langle\rangle) = \{1^\infty\}$$
$$\text{FairMerge } (1^\infty, \langle 2\rangle) = \{1^k \hat{}\langle 2\rangle\hat{}1^\infty : k \in \mathbb{N}\}$$

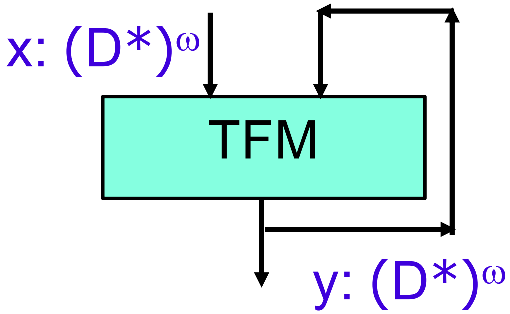
$$1^\infty \sqsubseteq 1^\infty \wedge \langle\rangle \sqsubseteq \langle 2\rangle$$

but

$$\text{FairMerge } (1^\infty, \langle\rangle) \cancel{\sqsubseteq} \text{ FairMerge } (1^\infty, \langle 2\rangle)$$

FairMerge is not prefix monotonic!

Feedback for Timed Fair Merge



$$TFM^{\circlearrowleft}(x) = \{y: y \in TFM(x, y)\}$$

Since TFM is fully realizable: if

$$y \in TFM^{\circlearrowleft}(x)$$

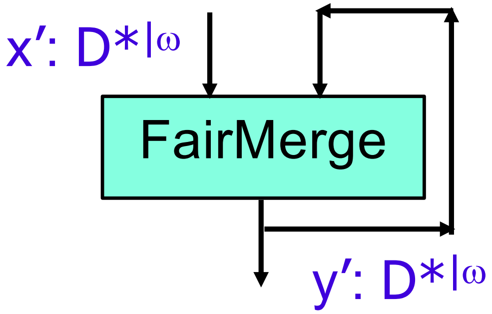
then there exists a realization

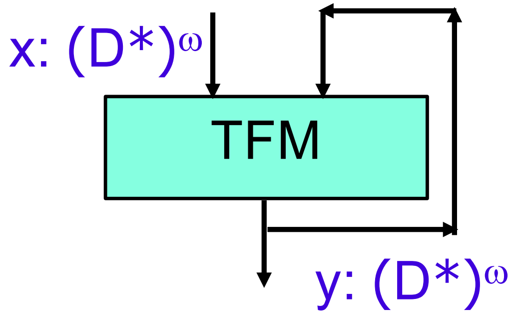
$$f: (D^*)^{\omega} \times (D^*)^{\omega} \rightarrow (D^*)^{\omega}$$

which has a unique fixpoint (Banach fixpoint)

$$y = f(x, y)$$

Feedback for Fair Merge



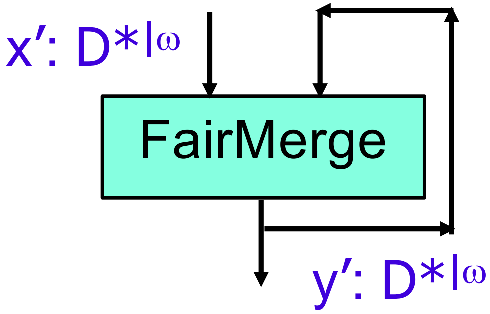


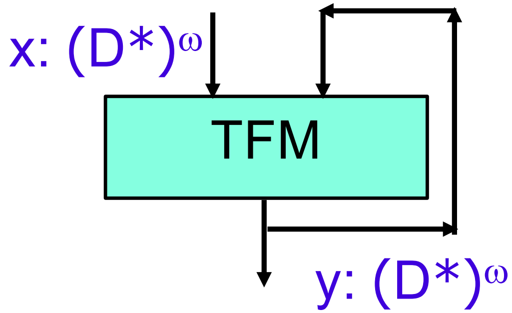
$$\bar{x} = \langle 1 \rangle \wedge y \in \mathrm{TFM}^{\circlearrowleft}(x) \Rightarrow \bar{y} = 1^{\infty}$$

$$y' \in \mathrm{FairMerge}(\langle 1 \rangle, y') \quad \cancel{\Rightarrow} \quad y' = 1^{\infty}$$

$$(2\hat{}1)^{\omega} \cancel{\in} \mathrm{TFM}^{\circlearrowleft}(\langle 1 \rangle)$$

Feedback for Fair Merge





Define

$$\mathrm{FairMerge}^{\circlearrowleft}(x') = \overline{\mathrm{TFM}^{\circlearrowleft}}(x')$$

Time is on my side …

The Rolling Stones

Generalization

The described techniques work

- for all fully realizable specifications which are time insensitive; they can be specified by describing their time abstraction
- concurrent composition with feedback applied to time abstractions of fully realizable specifications which are time insensitive leads time abstractions of to fully realizable specifications which are time insensitive
- deterministic monotonic functions on untimed streams are fully realizable specifications which are time insensitive

This way a fixpoint theory for time abstractions of of fully realizable specifications which are time insensitive is available!

Interrupt: Timed – Immediate Interrupt

Consider the function

$$\text{ITRPT}: (\{1\}^*)^\omega \times (D^*)^\omega \times (D^*)^\omega \rightarrow \wp(D^{*|\omega})$$

$$\text{ITRPT}(\varepsilon, x, z) = \{x\}$$

$$1\#(s\downarrow t) = 0 \wedge 1\#(s\downarrow t+1) > 0 \Rightarrow \text{ITRPT}(s, x, z) = \{\langle\langle\rangle\rangle\hat{\ }(x\downarrow t)\hat{\ }z\}$$

ITPRT is strongly causal and fully realizable, but not time insensitive!

Consider the function

$$\text{ITRPTD}: (\{1\}^*)^\omega \times (D^*)^\omega \times (D^*)^\omega \rightarrow \wp((D^*)^\omega)$$

$$\text{ITRPTD}(\varepsilon, x, z) = \{\langle\langle\rangle\rangle\hat{\ }x\}$$

$$1\#(s\downarrow t) = 0 \wedge 1\#(s\downarrow t+1) > 0 \Rightarrow \text{ITRPTD}(s, x, z) = \{\langle\langle\rangle\rangle\hat{\ }(x\downarrow(t+d))\hat{\ }z: d \in \mathbb{N}\}$$

Interrupt with delay

ITPRTD is strongly causal and fully realizable, but not time insensitive!

Consider the function

$$\text{ITRPTU}: \{1\}^{*|\omega} \times D^{*|\omega} \times D^{*|\omega} \to \wp(D^{*|\omega})$$

$$\text{ITRPTU}(\langle\rangle, x, z) = \{x\}$$

$$s \neq \langle\rangle \Rightarrow \text{ITRPTU}(s, x, z) = \{(x{\downarrow}t)\hat{\ }z : t \in \mathbb{N}\}$$

$$\langle\rangle \sqsubseteq \langle 1\rangle$$

but

$$\text{ITRPTU}(\langle\rangle, 2^\infty, 3^\infty) \cancel{\sqsubseteq} \text{ITRPTU}(\langle 1\rangle, 2^\infty, 3^\infty)$$

$$\text{ITRPTU}(\langle\rangle, 2^\infty, 3^\infty) = \{2^\infty\} \qquad 2^\infty \cancel{\in} \text{ITRPTU}(\langle 1\rangle, 2^\infty, 3^\infty)$$

Untimed interrupt is not monotonic!

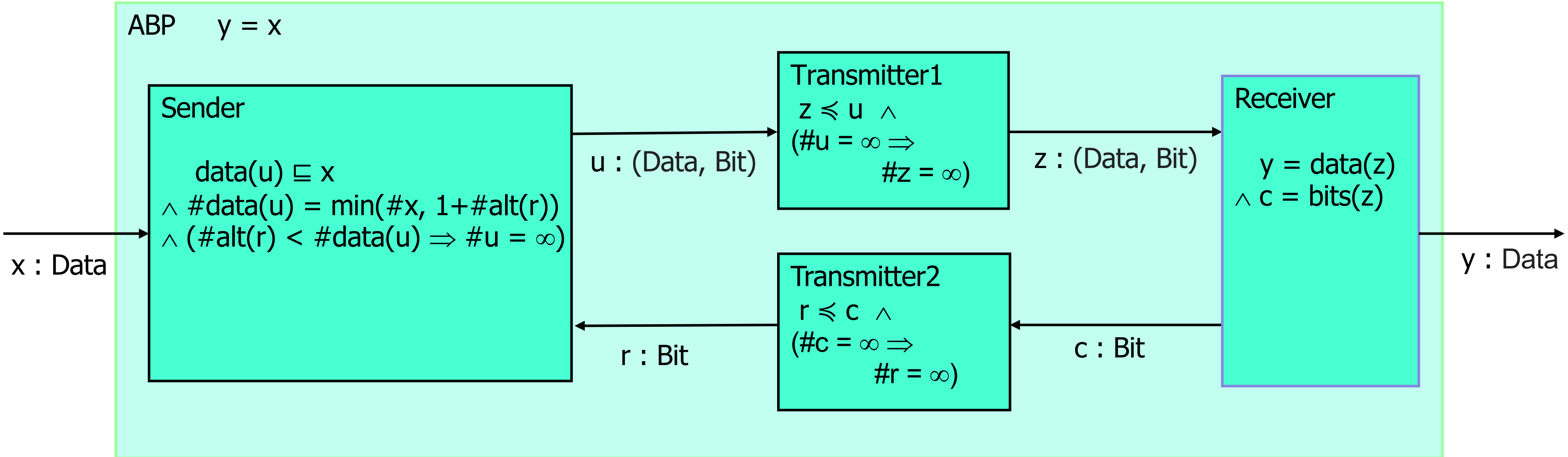
Feasible Abstractions

Define

$$\text{ITRPTU}(s, x, z) = \overline{\text{ITRPTD}}(s, x, z)$$

An untimed behavior is called feasible abstraction if it is the abstraction of a timed behavior which is fully realizable

Alternating Bit Protocol



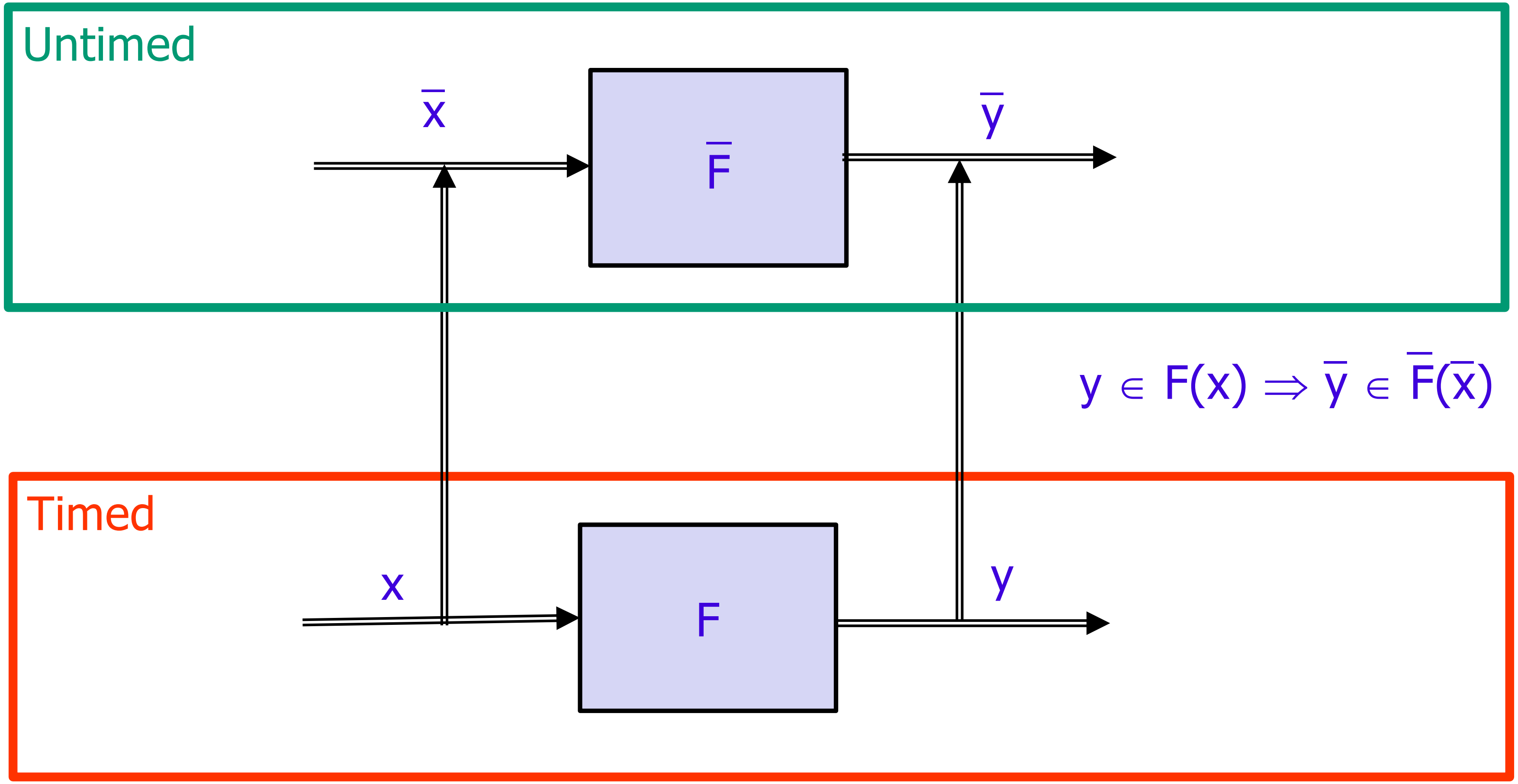
Two partial orderings on streams x, y are used :

Prefix: $x \sqsubseteq y = \exists\, z{:}\ x\hat{\ }z = y$

Substream: Let $\preccurlyeq$ the weakest relation which fulfills the the following equation

$$x \preccurlyeq y = (x = \langle\rangle \vee \exists\, n \in \mathbb{N}{:}\ (\text{first}(x) = \text{first}(\text{rest}^n(y)) \wedge \text{rest}(x) \preccurlyeq \text{rest}^{n+1}(y)\ ))$$

Levels of abstraction for time insensitive behaviors: untimed versus timed



Levels of abstraction

- Untimed behavior can always be understood as the result of an abstraction of timed behavior
- Timed behavior may be time insensitive, which means that time can be abstracted away since time abstraction models the untimed behavior
- Behaviors which are time insensitive and fully realizable with an abstract nondeterministic behavior which is not monotonic can be treated in composition with the help of their timed representation which is only a auxiliary construction and can be abstracted away afterwards

Bottom Line

If a nondeterministic untimed behavior F is the abstraction of a time insensitive behavior F′ which is fully realizable then (even if F is not monotonic)

- the data flow of F′ can be specified by the untimed specification for F
- the result of composition (including feedback) of untimed behavior can be determined at the level of timed behaviors and then abstracted into untimed behaviors