Nondeterministic Asynchronous Dataflow in Isabelle/HOL

Rafael Castro G. Silva, Laouen Fernet and Dmitriy Traytel

Department of Computer Science
University of Copenhagen

14/05/2025

Motivation

Motivation

Context:

- Stream Processing: programs that compute (possibly) unbounded sequences of data (streams)
- A common problem in the industry
- Frameworks: Apache Flink, Google Cloud Dataflow, and Timely Dataflow

- Why use frameworks?
  - Highly Parallel
  - Low latency (output as soon as possible)
  - Incremental computing (re-uses previous computations)

Motivation

Context:

- Stream Processing: programs that compute (possibly) unbounded sequences of data (streams)
- A common problem in the industry
- Frameworks: Apache Flink, Google Cloud Dataflow, and Timely Dataflow

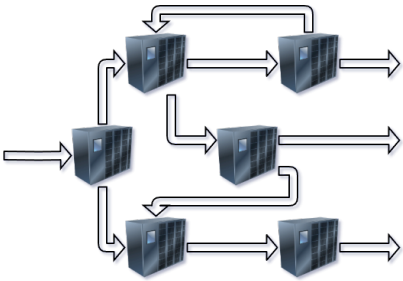
- Why use frameworks?
  - Highly Parallel
  - Low latency (output as soon as possible)
  - Incremental computing (re-uses previous computations)

Our goal:

Mechanically Verify Timely Dataflow algorithms

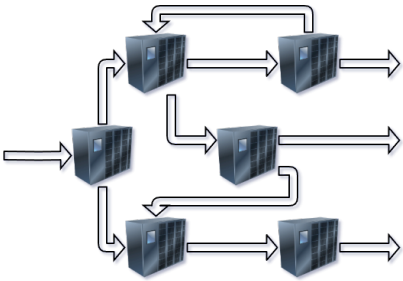
A Good Foundation

A Good Foundation



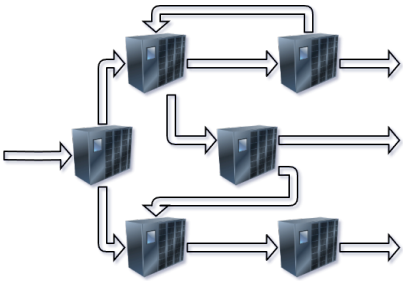
- Nondeterministic Asynchronous Dataflow

A Good Foundation



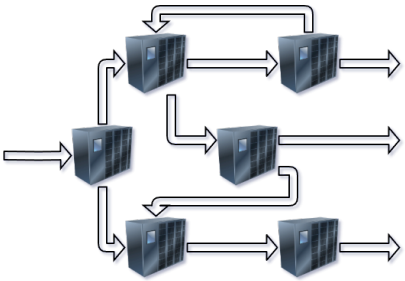
- Nondeterministic Asynchronous Dataflow
  - Dataflow: Directed graph of interconnected operators

A Good Foundation



- Nondeterministic Asynchronous Dataflow
  - Dataflow: Directed graph of interconnected operators
  - Asynchronous:
    - Operators execute independently: processes without an orchestrator
    - Operators can freely communicate with the network (read/write); do silent computation steps
    - Networks are unbounded FIFO queues

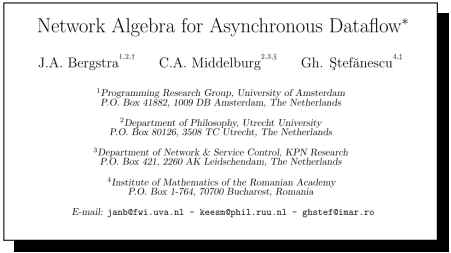
A Good Foundation



- Nondeterministic Asynchronous Dataflow
  - Dataflow: Directed graph of interconnected operators
  - Asynchronous:
    - Operators execute independently: processes without an orchestrator
    - Operators can freely communicate with the network (read/write); do silent computation steps
    - Networks are unbounded FIFO queues
  - Nondeterministic:
    - Operators can make nondeterministic choices

The Algebra for Nondeterministic Asynchronous Dataflow

- Bergstra et al. presents an algebra for Nondeterministic Asynchronous Dataflow
- Primitives:
  sequential and parallel composition; feedback loop...
- 52 axioms
- A process calculus instance



Network Algebra for Asynchronous Dataflow*

J.A. Bergstra[1,2,†] C.A. Middelburg[2,3,§] Gh. Ştefănescu[4,‡]

[1]Programming Research Group, University of Amsterdam
P.O. Box 41882, 1009 DB Amsterdam, The Netherlands

[2]Department of Philosophy, Utrecht University
P.O. Box 80126, 3508 TC Utrecht, The Netherlands

[3]Department of Network & Service Control, KPN Research
P.O. Box 421, 2260 AK Leidschendam, The Netherlands

[4]Institute of Mathematics of the Romanian Academy
P.O. Box 1-764, 70700 Bucharest, Romania

E-mail: janb@fwi.uva.nl - keesm@phil.ruu.nl - ghstef@imar.ro

Main Contributions

- A Isabelle/HOL instance of Nondeterministic Asynchronous Dataflow
  - Operators as a shallow embedding as codatatypes
  - 51 axioms proved
- Executable via code extraction to Haskell

Isabelle/HOL Preliminaries

Isabelle/HOL

- Classical higher-order logic (HOL):
  Simple Typed Lambda Calculus + axiom of choice + axiom of infinity + rank-1 polymorphism

Isabelle/HOL

- Classical higher-order logic (HOL):
  Simple Typed Lambda Calculus + axiom of choice + axiom of infinity + rank-1 polymorphism
- Isabelle: A generic proof assistant

- Isabelle/HOL: Isabelle's flavor of HOL

Isabelle/HOL

- Classical higher-order logic (HOL):
  Simple Typed Lambda Calculus + axiom of choice + axiom of infinity + rank-1 polymorphism
- Isabelle: A generic proof assistant

- Isabelle/HOL: Isabelle's flavor of HOL

Why Isabelle/HOL?

- Codatatypes: (possibly) infinite data structures (e.g., lazy lists, streams)
- Corecursion: always eventually produces some codatatype constructor
- Coinductive predicate: infinite number of introduction rule applications
- Coinduction: reason about coinductive predicates

Operators as a Codatatype

Operators

Operators in Isabelle/HOL

codatatype (inputs: $'i$, outputs: $'o$, $'d$) op =
Read $'i$ ($'d \Rightarrow$ ($'i$, $'o$, $'d$) op) | Write (($'i$, $'o$, $'d$) op) $'o$ $'d$
Silent ($'i$, $'o$, $'d$) op | Choice (($'i$, $'o$, $'d$) op) $cset$

Operators

Operators in Isabelle/HOL

codatatype (inputs: $'i$, outputs: $'o$, $'d$) op =
Read $'i$ ($'d \Rightarrow ('i, 'o, 'd)\ op$) | Write (($'i, 'o, 'd$) op) $'o$ $'d$
Silent ($'i, 'o, 'd$) op | Choice (($'i, 'o, 'd$) op) $cset$

- Type parameters:
  inputs/output ports; data
- Operator's actions
- Possibly infinite trees
- inputs/outputs:
  Sets of used ports

Operators

Operators in Isabelle/HOL

codatatype (inputs: $'i$, outputs: $'o$, $'d$) op =
Read $'i$ ($'d \Rightarrow$ ($'i$, $'o$, $'d$) op) | Write (($'i$, $'o$, $'d$) op) $'o$ $'d$
Silent ($'i$, $'o$, $'d$) op | Choice (($'i$, $'o$, $'d$) op) $cset$

- Type parameters: inputs/output ports; data
- Operator's actions
- Possibly infinite trees
- inputs/outputs: Sets of used ports

Uncommunicative operators

$\oslash$ = Choice $\{\}_c$

$\odot$ = Silent $\odot$

$\otimes$ = Choice $\{\otimes\}_c$

Operators

Operators in Isabelle/HOL

codatatype (inputs: $'i$, outputs: $'o$, $'d$) op =
Read $'i$ ($'d \Rightarrow ('i, 'o, 'd)\ op$) | Write (($'i, 'o, 'd$) op) $'o$ $'d$
Silent ($'i, 'o, 'd$) op | Choice (($'i, 'o, 'd$) op) $cset$

- Type parameters: inputs/output ports; data
- Operator's actions
- Possibly infinite trees
- inputs/outputs: Sets of used ports

Uncommunicative operators

$\oslash$ = Choice $\{\}_c$

$\odot$ = Silent $\odot$

$\otimes$ = Choice $\{\otimes\}_c$

More examples

ex1 = Choice $\{$Write ex1 1 42, $\oslash\}_c$

ex2 = Choice $\{$Write ex2 1 42, ex2$\}_c$

ex3 = Choice $\{$Write ex3 1 42, Silent ex3$\}_c$

Operators Equivalences: Motivation

An ideal equivalence relation

- Only equate operators that **behave** the same
- Useful reasoning principle

Operators Equivalences: Motivation

An ideal equivalence relation

- Only equate operators that **behave** the same
- Useful reasoning principle

- Milner's approach: Weak Bisimilarity

- Based on labeled transition systems (LTS)
- Labels (actions): Read, Write, Silent (τ)

Operators Equivalences: Label Transition System

Label Transition System

datatype $('i, 'o, 'd)\ IO = \text{Inp}\ 'i\ 'd \mid \text{Out}\ 'o\ 'd \mid \text{Tau}$

inductive step **where**

$\text{step}\ (\text{Inp}\ p\ x)\ (\text{Read}\ p\ f)\ (f\ x) \mid \text{step}\ (\text{Out}\ q\ x)\ (\text{Write}\ op\ q\ x)\ op$

$\mid \text{step}\ \text{Tau}\ (\text{Silent}\ op)\ op$

$\mid op \in_c ops \implies \text{step}\ io\ op\ op' \implies \text{step}\ io\ (\text{Choice}\ ops)\ op'$

Operators Equivalences: Weak Bisimilarity

Weakly Simulates

$$\text{estep Tau} = (\text{step Tau})^{==}$$

$$\text{estep } io = \text{step } io$$

$$\text{wstep } io = (\text{step Tau})^{**} \text{ OO } (\text{estep } io) \text{ OO } (\text{step Tau})^{**}$$

$$\text{wsim } R\ op_1\ op_2 = (\forall io\ op_1'.\ \text{step } io\ op_1\ op_1' \longrightarrow (\exists op_2'.\ \text{wstep } io\ op_2\ op_2' \wedge R\ op_1'\ op_2'))$$

Operators Equivalences: Weak Bisimilarity

Weakly Simulates

$$\text{estep Tau} = (\text{step Tau})^{==}$$

$$\text{estep } io = \text{step } io$$

$$\text{wstep } io = (\text{step Tau})^{**} \text{ OO } (\text{estep } io) \text{ OO } (\text{step Tau})^{**}$$

$$\text{wsim } R\ op_1\ op_2 = (\forall io\ op_1'.\ \text{step } io\ op_1\ op_1' \longrightarrow (\exists op_2'.\ \text{wstep } io\ op_2\ op_2' \wedge R\ op_1'\ op_2'))$$

Weak Bisimilarity

coinductive wbisim (**infix** $\approx$ 40) **where**

$$\text{wsim } (\approx)\ op_1\ op_2 \implies \text{wsim } (\approx)\ op_2\ op_1 \implies op_1 \approx op_2$$

Operators Equivalences: Weak Bisimilarity

Weakly Simulates

estep Tau $=$ (step Tau)$^{==}$

estep io $=$ step io

wstep io $=$ (step Tau)** OO (estep io) OO (step Tau)**

wsim R op_1 op_2 $= (\forall io\ op_1'.\ \text{step}\ io\ op_1\ op_1' \longrightarrow (\exists op_2'.\ \text{wstep}\ io\ op_2\ op_2' \wedge R\ op_1'\ op_2'))$

Weak Bisimilarity

coinductive wbisim (**infix** $\approx$ 40) **where**

wsim ($\approx$) op_1 op_2 $\implies$ wsim ($\approx$) op_2 op_1 $\implies$ $op_1 \approx op_2$

- $\approx$ has a useful coinduction principle
- $\oslash \approx \otimes \approx \odot$
- ex1 $\approx$ ex2 $\approx$ ex3

Asynchronous Dataflow Operators

Buffer Infrastructure

- foo

Asynchronous Dataflow Properties

Basic network algebra properties

B1: $op_1 \parallel (op_2 \parallel op_3) \approx \mathsf{map\_op}\ \curvearrowright\ \curvearrowright\ (op_1 \parallel op_2) \parallel op_3$

B2_1: $op \parallel (\mathcal{I} :: (0, 0, {}'d)\ op) \approx \mathsf{map\_op}\ \mathsf{Inl}\ \mathsf{Inl}\ op$

B2_2: $(\mathcal{I} :: (0, 0, {}'d)\ op) \parallel op \approx \mathsf{map\_op}\ \mathsf{Inr}\ \mathsf{Inr}\ op$

B3: $(op_1 \bullet op_2) \bullet op_3 \approx op_1 \bullet (op_2 \bullet op_3)$

B4_1: $op\vdash \bullet \mathcal{I} \approx op\vdash$

B4_2: $\mathcal{I} \bullet \dashv op \approx \dashv op$

B5: $(op_1 \parallel op_2) \bullet (op_3 \parallel op_4) \approx (op_1 \bullet op_3) \parallel (op_2 \bullet op_4)$

B6: $\mathcal{I} \parallel \mathcal{I} \approx \mathcal{I}$

B7: $\mathcal{X} \bullet \mathcal{X} \approx \mathcal{I}$

B8: $(\mathcal{X} :: ({}'i + 0, 0 + {}'i, {}'d)\ op) \approx \mathsf{map\_op}\ \mathsf{id}\ (\mathsf{case\_sum}\ \mathsf{Inr}\ \mathsf{Inl})\ \mathcal{I}$

B9: $\mathcal{X} \approx \mathsf{map\_op}\ \curvearrowright\ \curvearrowright\ (\mathcal{X} \parallel \mathcal{I}) \bullet \mathsf{map\_op}\ \mathsf{id}\ \curvearrowleft\ (\mathcal{I} \parallel \mathcal{X})$

B10: $(\dashv op_1 \parallel \dashv op_2) \bullet \mathcal{X} \approx \mathcal{X} \bullet (op_2\vdash \parallel op_1\vdash)$

F1: $\mathcal{I}\uparrow \approx (\mathcal{I} :: (0, 0, {}'d)\ op)$

F2: $\mathcal{X}\uparrow \approx \mathcal{I}$

Table 1 Basic network algebra properties

R1: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow op_2 \bullet (op_1\uparrow) \approx ((op_2 \parallel \mathcal{I}) \bullet op_1)\uparrow$

R2: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow (op_1\uparrow) \bullet op_2 \approx (op_1 \bullet (op_2 \parallel \mathcal{I}))\uparrow$

R3: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow op_1 \parallel (op_2\uparrow) \approx (\mathsf{map\_op}\ \curvearrowleft\ \curvearrowleft\ (op_1 \parallel op_2))\uparrow$

R4: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{inputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{outputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow (\dashv op_1 \bullet (\mathcal{I} \parallel op_2))\uparrow \approx ((\mathcal{I} \parallel op_2) \bullet op_1\vdash)\uparrow$

R5: $\mathsf{Inr} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow \mathsf{map\_op}\ \mathsf{Inl}\ \mathsf{Inl}\ ((op :: ({}'i + 0, {}'o + 0, {}'d)\ op)\uparrow) \approx op$

R6: $\mathsf{Inr} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{Inl} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{Inl} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow (op\uparrow)\uparrow \approx (\mathsf{map\_op}\ \curvearrowright\ \curvearrowright\ op)\uparrow$

Basic network algebra properties

B1: $op_1 \parallel (op_2 \parallel op_3) \approx \mathsf{map\_op}\ \frown\ \frown\ (op_1 \parallel op_2) \parallel op_3$

B2_1: $op \parallel (\mathcal{I} :: (0, 0, {}'d)\ op) \approx \mathsf{map\_op\ Inl\ Inl}\ op$

B2_2: $(\mathcal{I} :: (0, 0, {}'d)\ op) \parallel op \approx \mathsf{map\_op\ Inr\ Inr}\ op$

B3: $(op_1 \bullet op_2) \bullet op_3 \approx op_1 \bullet (op_2 \bullet op_3)$

B4_1: $op\vdash \bullet \mathcal{I} \approx op\vdash$

B4_2: $\mathcal{I} \bullet \dashv op \approx \dashv op$

B5: $(op_1 \parallel op_2) \bullet (op_3 \parallel op_4) \approx (op_1 \bullet op_3) \parallel (op_2 \bullet op_4)$

B6: $\mathcal{I} \parallel \mathcal{I} \approx \mathcal{I}$

B7: $\mathcal{X} \bullet \mathcal{X} \approx \mathcal{I}$

B8: $(\mathcal{X} :: ({}'i + 0, 0 + {}'i, {}'d)\ op) \approx \mathsf{map\_op\ id\ (case\_sum\ Inr\ Inl)}\ \mathcal{I}$

B9: $\mathcal{X} \approx \mathsf{map\_op}\ \frown\ \frown\ (\mathcal{X} \parallel \mathcal{I}) \bullet \mathsf{map\_op\ id}\ \curvearrowleft\ (\mathcal{I} \parallel \mathcal{X})$

B10: $(\dashv op_1 \parallel \dashv op_2) \bullet \mathcal{X} \approx \mathcal{X} \bullet (op_2\vdash \parallel op_1\vdash)$

F1: $\mathcal{I}\uparrow \approx (\mathcal{I} :: (0, 0, {}'d)\ op)$

F2: $\mathcal{X}\uparrow \approx \mathcal{I}$

Table 1 Basic network algebra properties

R1: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow op_2 \bullet (op_1\uparrow) \approx ((op_2 \parallel \mathcal{I}) \bullet op_1)\uparrow$

R2: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow (op_1\uparrow) \bullet op_2 \approx (op_1 \bullet (op_2 \parallel \mathcal{I}))\uparrow$

R3: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow op_1 \parallel (op_2\uparrow) \approx (\mathsf{map\_op}\ \curvearrowleft\ \curvearrowleft\ (op_1 \parallel op_2))\uparrow$

R4: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{inputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{outputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow (\dashv op_1 \bullet (\mathcal{I} \parallel op_2))\uparrow \approx ((\mathcal{I} \parallel op_2) \bullet op_1\vdash)\uparrow$

R5: $\mathsf{Inr} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow \mathsf{map\_op\ Inl\ Inl}\ ((op :: ({}'i + 0, {}'o + 0, {}'d)\ op)\uparrow) \approx op$

R6: $\mathsf{Inr} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{Inl} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{Inl} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow (op\uparrow)\uparrow \approx (\mathsf{map\_op}\ \frown\ \frown\ op)\uparrow$

Basic network algebra properties

B1: $op_1 \parallel (op_2 \parallel op_3) \approx \mathsf{map\_op}\ \curvearrowright\ \curvearrowright\ (op_1 \parallel op_2) \parallel op_3$

B2_1: $op \parallel (\mathcal{I} :: (0, 0, {}'d)\ op) \approx \mathsf{map\_op}\ \mathsf{Inl}\ \mathsf{Inl}\ op$

B2_2: $(\mathcal{I} :: (0, 0, {}'d)\ op) \parallel op \approx \mathsf{map\_op}\ \mathsf{Inr}\ \mathsf{Inr}\ op$

B3: $(op_1 \bullet op_2) \bullet op_3 \approx op_1 \bullet (op_2 \bullet op_3)$

B4_1: $op\vdash \bullet \mathcal{I} \approx op\vdash$

B4_2: $\mathcal{I} \bullet \dashv op \approx \dashv op$

B5: $(op_1 \parallel op_2) \bullet (op_3 \parallel op_4) \approx (op_1 \bullet op_3) \parallel (op_2 \bullet op_4)$

B6: $\mathcal{I} \parallel \mathcal{I} \approx \mathcal{I}$

B7: $\mathcal{X} \bullet \mathcal{X} \approx \mathcal{I}$

B8: $(\mathcal{X} :: ({}'i + 0, 0 + {}'i, {}'d)\ op) \approx \mathsf{map\_op}\ \mathsf{id}\ (\mathsf{case\_sum}\ \mathsf{Inr}\ \mathsf{Inl})\ \mathcal{I}$

B9: $\mathcal{X} \approx \mathsf{map\_op}\ \curvearrowright\ \curvearrowright\ (\mathcal{X} \parallel \mathcal{I}) \bullet \mathsf{map\_op}\ \mathsf{id}\ \curvearrowleft\ (\mathcal{I} \parallel \mathcal{X})$

B10: $(\dashv op_1 \parallel \dashv op_2) \bullet \mathcal{X} \approx \mathcal{X} \bullet (op_2\vdash \parallel op_1\vdash)$

F1: $\mathcal{I}\uparrow \approx (\mathcal{I} :: (0, 0, {}'d)\ op)$

F2: $\mathcal{X}\uparrow \approx \mathcal{I}$

Table 1 Basic network algebra properties

R1: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow op_2 \bullet (op_1\uparrow) \approx ((op_2 \parallel \mathcal{I}) \bullet op_1)\uparrow$

R2: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow (op_1\uparrow) \bullet op_2 \approx (op_1 \bullet (op_2 \parallel \mathcal{I}))\uparrow$

R3: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow op_1 \parallel (op_2\uparrow) \approx (\mathsf{map\_op}\ \curvearrowleft\ \curvearrowleft\ (op_1 \parallel op_2))\uparrow$

R4: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{inputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{outputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow (\dashv op_1 \bullet (\mathcal{I} \parallel op_2))\uparrow \approx ((\mathcal{I} \parallel op_2) \bullet op_1\vdash)\uparrow$

R5: $\mathsf{Inr} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow \mathsf{map\_op}\ \mathsf{Inl}\ \mathsf{Inl}\ ((op :: ({}'i + 0, {}'o + 0, {}'d)\ op)\uparrow) \approx op$

R6: $\mathsf{Inr} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{Inl} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{Inl} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow (op\uparrow)\uparrow \approx (\mathsf{map\_op}\ \curvearrowright\ \curvearrowright\ op)\uparrow$

Basic network algebra properties

B1: $op_1 \parallel (op_2 \parallel op_3) \approx \mathsf{map\_op}\ \curvearrowright\ \curvearrowright\ (op_1 \parallel op_2) \parallel op_3$

B2_1: $op \parallel (\mathcal{I} :: (0, 0, {}'d)\ op) \approx \mathsf{map\_op}\ \mathsf{Inl}\ \mathsf{Inl}\ op$

B2_2: $(\mathcal{I} :: (0, 0, {}'d)\ op) \parallel op \approx \mathsf{map\_op}\ \mathsf{Inr}\ \mathsf{Inr}\ op$

B3: $(op_1 \bullet op_2) \bullet op_3 \approx op_1 \bullet (op_2 \bullet op_3)$

B4_1: $op\vdash \bullet \mathcal{I} \approx op\vdash$

B4_2: $\mathcal{I} \bullet \dashv op \approx \dashv op$

B5: $(op_1 \parallel op_2) \bullet (op_3 \parallel op_4) \approx (op_1 \bullet op_3) \parallel (op_2 \bullet op_4)$

B6: $\mathcal{I} \parallel \mathcal{I} \approx \mathcal{I}$

B7: $\mathcal{X} \bullet \mathcal{X} \approx \mathcal{I}$

B8: $(\mathcal{X} :: ({}'i + 0, 0 + {}'i, {}'d)\ op) \approx \mathsf{map\_op}\ \mathsf{id}\ (\mathsf{case\_sum}\ \mathsf{Inr}\ \mathsf{Inl})\ \mathcal{I}$

B9: $\mathcal{X} \approx \mathsf{map\_op}\ \curvearrowright\ \curvearrowright\ (\mathcal{X} \parallel \mathcal{I}) \bullet \mathsf{map\_op}\ \mathsf{id}\ \curvearrowleft\ (\mathcal{I} \parallel \mathcal{X})$

B10: $(\dashv op_1 \parallel \dashv op_2) \bullet \mathcal{X} \approx \mathcal{X} \bullet (op_2\vdash \parallel op_1\vdash)$

F1: $\mathcal{I}\uparrow \approx (\mathcal{I} :: (0, 0, {}'d)\ op)$

F2: $\mathcal{X}\uparrow \approx \mathcal{I}$

Table 1 Basic network algebra properties

R1: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow op_2 \bullet (op_1\uparrow) \approx ((op_2 \parallel \mathcal{I}) \bullet op_1)\uparrow$

R2: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow (op_1\uparrow) \bullet op_2 \approx (op_1 \bullet (op_2 \parallel \mathcal{I}))\uparrow$

R3: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow op_1 \parallel (op_2\uparrow) \approx (\mathsf{map\_op}\ \curvearrowleft\ \curvearrowleft\ (op_1 \parallel op_2))\uparrow$

R4: $\mathsf{Inr} \succeq \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{inputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{outputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow (\dashv op_1 \bullet (\mathcal{I} \parallel op_2))\uparrow \approx ((\mathcal{I} \parallel op_2) \bullet op_1\vdash)\uparrow$

R5: $\mathsf{Inr} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow \mathsf{map\_op}\ \mathsf{Inl}\ \mathsf{Inl}\ ((op :: ({}'i + 0, {}'o + 0, {}'d)\ op)\uparrow) \approx op$

R6: $\mathsf{Inr} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{Inl} \succeq \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr} \succeq \mathsf{Inl} \succeq \mathsf{outputs}\ op = \{\} \Longrightarrow (op\uparrow)\uparrow \approx (\mathsf{map\_op}\ \curvearrowright\ \curvearrowright\ op)\uparrow$

Conclusion

Conclusion

- Isabelle/HOL has a good tool set to formalize and reason about stream processing:
  - Codatatypes, coinductive predicates, corecursion with friends, reasoning up to friends (congruence),
    - Coinduction up to congruence principle is automatically derived for codatatypes (but not for coinductive principles)
- Next step: Feedback loop

Questions, comments and suggestions