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, Apache Samza, Apache Spark, Google Cloud Dataflow, and Timely Dataflow

Cloud DataFlow Flink

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

Motivation

Context:

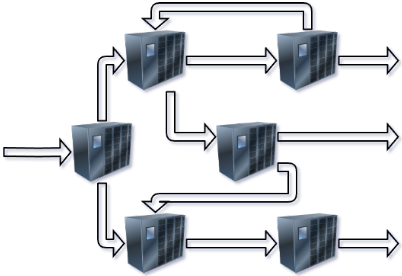
- Stream Processing: programs that compute (possibly) unbounded sequences of data (streams)
- A common problem in the industry
- Frameworks:
  Apache Flink, Apache Samza, Apache Spark, Google Cloud Dataflow, and Timely Dataflow
- Why use frameworks?
  - Highly Parallel
  - Low latency (output as soon as possible)
  - Incremental computing (re-uses previous computations)

Our goal:

Mechanically Verify Timely Dataflow algorithms

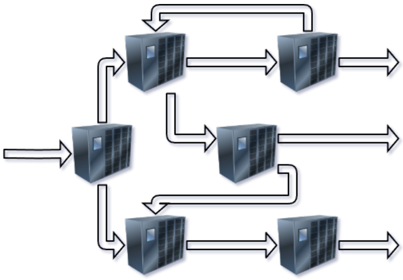
A Good Foundation

A Good Foundation



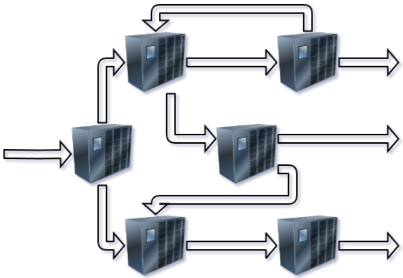
- Nondeterministic Asynchronous Dataflow

A Good Foundation



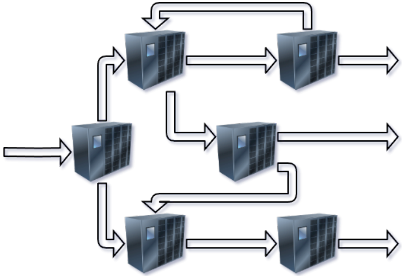
- Nondeterministic Asynchronous Dataflow
  - Dataflow: Directed graph of interconnected operators

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

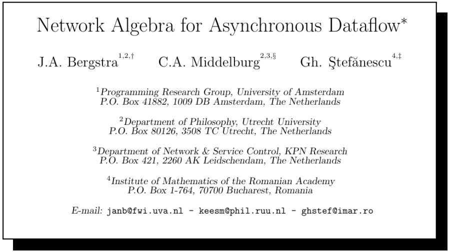
A Good Foundation



- 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
    - Operators are relations between inputs and outputs sequences

The Algebra for Nondeterministic Asynchronous Dataflow

- Bergstra et al. presents an algebra for Nondeterministic Asynchronous Dataflow
- Primitives:
  sequential and parallel composition; feedback loop...
- The 52 axioms
- An 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

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

$$\textbf{codatatype}\ (\text{inputs}: 'i, \text{outputs}: 'o, 'd)\ op =$$
$$\text{Read}\ 'i\ ('d \Rightarrow ('i, 'o, 'd)\ op) \mid \text{Write}\ (('i, 'o, 'd)\ op)\ 'o\ 'd$$
$$\text{Silent}\ ('i, 'o, 'd)\ op \mid \text{Choice}\ (('i, 'o, 'd)\ op)\ cset$$

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

Examples 1

Uncommunicative operators

abbreviation

$\oslash \equiv \mathsf{Choice}\ \{\}_c$

corec silent_op ($\odot$) **where**

$\odot = \mathsf{Silent}\ \odot$

corec spin_op ($\otimes$) **where**

$\otimes = \mathsf{Choice}\ ((\lambda\_.\ \otimes)\grave{}_c\ \{()\}_c)$

lemma spin_op_code:

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

- Quirk: any corecursive call guarded by the Choice must be applied using the proper map function
- They have the same meaning.

Examples 1

Uncommunicative operators

abbreviation
$\oslash \equiv \text{Choice } \{\}_c$

corec silent_op ($\odot$) **where**
$\odot = \text{Silent } \odot$

corec spin_op ($\otimes$) **where**
$\otimes = \text{Choice } ((\lambda\_.\ \otimes)`_c \{()\}_c)$

lemma spin_op_code:
$\otimes = \text{Choice } \{\otimes\}_c$

- Quirk: any corecursive call guarded by the Choice must be applied using the proper map function
- They have the same meaning. But are syntactically different ☹

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

Intuition

Two operators are bisimilar if their corresponding transition systems can mutually simulate each other's transitions.

Operators Equivalences: Label Transition System

Label Transition System

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

$\textbf{inductive}\ \mathsf{step}\ \textbf{where}$

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

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

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

Operators Equivalences: Strong Bisimilarity

Simulates

definition sim where

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

Strong Bisimilarity

Operators Equivalences: Weak Bisimilarity

- foo

Asynchronous Dataflow Operators

Buffer Infrastructure

- foo

Asynchronous Dataflow Properties

Basic network algebra properties

$$\mathrm{B1}\colon op_1 \parallel (op_2 \parallel op_3) \approx \mathsf{map\_op}\ \curvearrowright \curvearrowright\ (op_1 \parallel op_2) \parallel op_3$$
$$\mathrm{B2\_1}\colon op \parallel (\mathcal{I} :: (\mathit{0}, \mathit{0}, 'd)\ op) \approx \mathsf{map\_op}\ \mathsf{Inl}\ \mathsf{Inl}\ op$$
$$\mathrm{B2\_2}\colon (\mathcal{I} :: (\mathit{0}, \mathit{0}, 'd)\ op) \parallel op \approx \mathsf{map\_op}\ \mathsf{Inr}\ \mathsf{Inr}\ op$$
$$\mathrm{B3}\colon (op_1 \bullet op_2) \bullet op_3 \approx op_1 \bullet (op_2 \bullet op_3)$$
$$\mathrm{B4\_1}\colon op{\vdash} \bullet \mathcal{I} \approx op{\vdash} \qquad \mathrm{B4\_2}\colon \mathcal{I} \bullet {\dashv}op \approx {\dashv}op$$
$$\mathrm{B5}\colon (op_1 \parallel op_2) \bullet (op_3 \parallel op_4) \approx (op_1 \bullet op_3) \parallel (op_2 \bullet op_4)$$
$$\mathrm{B6}\colon \mathcal{I} \parallel \mathcal{I} \approx \mathcal{I} \qquad \mathrm{B7}\colon \mathcal{X} \bullet \mathcal{X} \approx \mathcal{I}$$
$$\mathrm{B8}\colon (\mathcal{X} :: ('i + \mathit{0}, \mathit{0} + 'i, 'd)\ op) \approx \mathsf{map\_op}\ \mathsf{id}\ (\mathsf{case\_sum}\ \mathsf{Inr}\ \mathsf{Inl})\ \mathcal{I}$$
$$\mathrm{B9}\colon \mathcal{X} \approx \mathsf{map\_op}\ \curvearrowright \curvearrowright\ (\mathcal{X} \parallel \mathcal{I}) \bullet \mathsf{map\_op}\ \mathsf{id}\ \curvearrowleft\ (\mathcal{I} \parallel \mathcal{X})$$
$$\mathrm{B10}\colon ({\dashv}op_1 \parallel {\dashv}op_2) \bullet \mathcal{X} \approx \mathcal{X} \bullet (op_2{\vdash} \parallel op_1{\vdash})$$
$$\mathrm{F1}\colon \mathcal{I}\uparrow \approx (\mathcal{I} :: (\mathit{0}, \mathit{0}, 'd)\ op) \qquad \mathrm{F2}\colon \mathcal{X}\uparrow \approx \mathcal{I}$$

Table 1 Basic network algebra properties

$$\mathrm{R1}\colon \mathsf{Inr}\ -\grave{}\ \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr}\ -\grave{}\ \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow$$
$$op_2 \bullet (op_1\uparrow) \approx ((op_2 \parallel \mathcal{I}) \bullet op_1)\uparrow$$
$$\mathrm{R2}\colon \mathsf{Inr}\ -\grave{}\ \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr}\ -\grave{}\ \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow$$
$$(op_1\uparrow) \bullet op_2 \approx (op_1 \bullet (op_2 \parallel \mathcal{I}))\uparrow$$
$$\mathrm{R3}\colon \mathsf{Inr}\ -\grave{}\ \mathsf{inputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr}\ -\grave{}\ \mathsf{outputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow$$
$$op_1 \parallel (op_2\uparrow) \approx (\mathsf{map\_op}\ \curvearrowleft \curvearrowleft\ (op_1 \parallel op_2))\uparrow$$
$$\mathrm{R4}\colon \mathsf{Inr}\ -\grave{}\ \mathsf{inputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{Inr}\ -\grave{}\ \mathsf{outputs}\ op_1 \cap \mathsf{defaults} = \{\} \Longrightarrow$$
$$\mathsf{inputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow \mathsf{outputs}\ op_2 \cap \mathsf{defaults} = \{\} \Longrightarrow$$
$$({\dashv}op_1 \bullet (\mathcal{I} \parallel op_2))\uparrow \approx ((\mathcal{I} \parallel op_2) \bullet op_1{\vdash})\uparrow$$
$$\mathrm{R5}\colon \mathsf{Inr}\ -\grave{}\ \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr}\ -\grave{}\ \mathsf{outputs}\ op = \{\} \Longrightarrow$$
$$\mathsf{map\_op}\ \mathsf{Inl}\ \mathsf{Inl}\ ((op :: ('i + \mathit{0}, 'o + \mathit{0}, 'd)\ op)\uparrow) \approx op$$
$$\mathrm{R6}\colon \mathsf{Inr}\ -\grave{}\ \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr}\ -\grave{}\ \mathsf{outputs}\ op = \{\} \Longrightarrow$$
$$\mathsf{Inr}\ -\grave{}\ \mathsf{Inl}\ -\grave{}\ \mathsf{inputs}\ op = \{\} \Longrightarrow \mathsf{Inr}\ -\grave{}\ \mathsf{Inl}\ -\grave{}\ \mathsf{outputs}\ op = \{\} \Longrightarrow$$
$$(op\uparrow)\uparrow \approx (\mathsf{map\_op}\ \curvearrowright \curvearrowright\ op)\uparrow$$

Basic network algebra properties

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

$\mathrm{B2\_1}: op \parallel (\mathcal{I} :: (0, 0, 'd)\ op) \approx \mathsf{map\_op\ Inl\ Inl}\ op$

$\mathrm{B2\_2}: (\mathcal{I} :: (0, 0, 'd)\ op) \parallel op \approx \mathsf{map\_op\ Inr\ Inr}\ op$

$\mathrm{B3}: (op_1 \bullet op_2) \bullet op_3 \approx op_1 \bullet (op_2 \bullet op_3)$

$\mathrm{B4\_1}: op\vdash \bullet \mathcal{I} \approx op\vdash$ $\quad$ $\mathrm{B4\_2}: \mathcal{I} \bullet \dashv op \approx \dashv op$

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

$\mathrm{B6}: \mathcal{I} \parallel \mathcal{I} \approx \mathcal{I}$ $\quad$ $\mathrm{B7}: \mathcal{X} \bullet \mathcal{X} \approx \mathcal{I}$

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

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

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

$\mathrm{F1}: \mathcal{I} \uparrow \approx (\mathcal{I} :: (0, 0, 'd)\ op)$ $\quad$ $\mathrm{F2}: \mathcal{X} \uparrow \approx \mathcal{I}$

Table 1 Basic network algebra properties

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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