A Domain-Specific Language for Incremental and Modular Design of Large-Scale Verifiably-Safe Flow Networks

Assaf Kfoury

Boston University

October 2011
Syntax: examples of small flow networks

\[ F \text{ ("fork") } \]
\[ M \text{ ("merge") } \]
\[ \text{hole } X \]
**Syntax:** examples of **small** flow networks

\[ F ("fork") \quad M ("merge") \quad \text{hole } X \]

\[ \mathcal{A} \quad \mathcal{B} \]
Syntax: examples of small flow networks

F ("fork")

M ("merge")

hole X

small = fully specified, not to be assembled, amenable to “whole-system” analysis
Syntax: assembling flow networks

\[ F \oplus X \oplus X \oplus M \]
Syntax: assembling flow networks

graphic representation of

**incremental** and **modular** design and analysis
Syntax: assembling flow networks

graphic representation of **incremental** and **modular** design and analysis

\[ A \quad B \quad A \]
**Syntax:** assembling flow networks

Graphic representation of **incremental** and **modular** design and analysis

\[ A \quad B \quad A \]
Syntax: assembling flow networks

graphic representation of incremental and modular design and analysis

\[ A \oplus B \quad A \]
Syntax: assembling flow networks

graphic representation of incremental and modular design and analysis

\[ A \oplus B \oplus A \]
Syntax: assembling flow networks

let $X \in \{A, B\}$ in \((F \oplus X \oplus X \oplus M)\)
**Syntax:** assembling flow networks

\[
\text{let } X \in \{A, B\} \text{ in } \left( F \oplus X \oplus X \oplus M \right)
\]

Different possible interpretations/reductions of (non-recursive) let-in:

- \[ F \oplus A \oplus A \oplus M \] \hspace{1cm} \text{OR} \hspace{1cm} \[ F \oplus B \oplus B \oplus M \]
Syntax: assembling flow networks

\[
\text{let } X \in \{ A, B \} \text{ in } (F \oplus X \oplus X \oplus M)
\]

Different possible interpretations/reductions of (non-recursive) let-in:

- \( F \oplus A \oplus A \oplus M \) \text{ OR } \( F \oplus B \oplus B \oplus M \)

- \( F \oplus A \oplus A \oplus M \) \text{ AND } \( F \oplus B \oplus B \oplus M \)
**Syntax: assembling flow networks**

\[
\text{let } X \in \{A, B\} \text{ in } (F \oplus X \oplus X \oplus M)
\]

Different possible interpretations/reductions of (non-recursive) let-in:

- \( F \oplus A \oplus A \oplus M \) **OR** \( F \oplus B \oplus B \oplus M \)

- \( F \oplus A \oplus A \oplus M \) **AND** \( F \oplus B \oplus B \oplus M \)

- \( F \oplus A \oplus A \oplus M \) **AND** \( F \oplus B \oplus B \oplus M \) **AND**
  \( F \oplus A \oplus B \oplus M \) **AND** \( F \oplus B \oplus A \oplus M \)
Syntax: assembling flow networks

\[
\text{let } X \in \{A, B\} \text{ in } (F \oplus X \oplus X \oplus M)
\]

Different possible interpretations/reductions of (non-recursive) let-in:

- \(F \oplus A \oplus A \oplus M\) \text{ OR } \(F \oplus B \oplus B \oplus M\)

- \(F \oplus A \oplus A \oplus M\) \text{ AND } \(F \oplus B \oplus B \oplus M\) (our interpretation here!)

- \(F \oplus A \oplus A \oplus M\) \text{ AND } \(F \oplus B \oplus B \oplus M\) \text{ AND }
  \(F \oplus A \oplus B \oplus M\) \text{ AND } \(F \oplus B \oplus A \oplus M\)
In this presentation, we take the network specification

\[
\text{let } X \in \{A, B\} \text{ in } \left( F \oplus X \oplus X \oplus M \right)
\]

as equivalent to

\[
\text{let } X = A \text{ in } \\
\text{let } Y = B \text{ in } \\
\left( \left( F \oplus X \oplus X \oplus M \right) \parallel \left( F \oplus Y \oplus Y \oplus M \right) \right)
\]
Syntax: assembling flow networks

In this presentation, we take the network specification

\[
\text{let } X \in \{A, B\} \text{ in } \left( F \oplus X \oplus X \oplus M \right)
\]

as equivalent to

\[
\text{let } X = A \text{ in } \\
\text{let } Y = B \text{ in } \\
\left( (F \oplus X \oplus X \oplus M) \parallel (F \oplus Y \oplus Y \oplus M) \right)
\]

The **semantics** and the **typings** will be defined to guarantee that both sides of the parallel constructor “\(\parallel\)” are “safe” to use.
Syntax: another construction beyond our current DSL

\[
\text{letrec } X \in \{ \mathcal{A} \oplus X, \mathcal{B} \oplus X \} \text{ in } \left( F \oplus X \oplus M \right)
\]
Syntax: another construction beyond our current DSL

\[ \text{letrec } X \in \{ A \oplus X, B \oplus X \} \text{ in } (F \oplus X \oplus M) \]

Different possible interpretations of letrec-in:
Syntax: another construction beyond our current DSL

```
letrec X ∈ {A ⊕ X, B ⊕ X} in (F ⊕ X ⊕ M)
```

Different possible interpretations of `letrec-in`:

- `F ⊕ A ⊕ A ⊕ A ⊕ ... ⊕ M` AND `F ⊕ B ⊕ B ⊕ B ⊕ ... ⊕ M`
**Syntax:** another construction beyond our current DSL

\[
\text{letrec } X \in \{ \mathcal{A} \oplus X, \mathcal{B} \oplus X \} \text{ in } \left( F \oplus X \oplus M \right)
\]

Different possible interpretations of \texttt{letrec-in}:

- \( F \oplus \mathcal{A} \oplus \mathcal{A} \oplus \mathcal{A} \oplus \cdots \oplus M \) AND \( F \oplus \mathcal{B} \oplus \mathcal{B} \oplus \mathcal{B} \oplus \cdots \oplus M \)

- \( \{ F \oplus \mathcal{A} \oplus M, \ F \oplus \mathcal{A} \oplus \mathcal{A} \oplus M, \ F \oplus \mathcal{A} \oplus \mathcal{A} \oplus \mathcal{A} \oplus M, \ \cdots \} \) AND \( \{ F \oplus \mathcal{B} \oplus M, \ F \oplus \mathcal{B} \oplus \mathcal{B} \oplus M, \ F \oplus \mathcal{B} \oplus \mathcal{B} \oplus \mathcal{B} \oplus M, \ \cdots \} \)
**Syntax:** another construction beyond our current DSL

\[
\text{letrec } X \in \{ A \oplus X, B \oplus X \} \text{ in } \left( F \oplus X \oplus M \right)
\]

Different possible interpretations of **letrec-in**:

- \( F \oplus A \oplus A \oplus A \oplus \cdots \oplus M \) AND \( F \oplus B \oplus B \oplus B \oplus \cdots \oplus M \)

- \( \left\{ F \oplus A \oplus M, \ F \oplus A \oplus A \oplus M, \ F \oplus A \oplus A \oplus A \oplus M, \ \cdots \right\} \)

AND

- \( \left\{ F \oplus B \oplus M, \ F \oplus B \oplus B \oplus M, \ F \oplus B \oplus B \oplus B \oplus M, \ \cdots \right\} \)

- \( \cdots \)
Syntax: formal specification of flow networks

\[ A, B \in \text{SMALLNETWORK} \]
\[ X, Y \in \text{HOLE} \]
\[ M, N \in \text{NETWORK} \]

\[ M, N \in \text{NETWORK} \quad ::= \quad A \quad \text{small network} \]
\[ X \quad \text{hole} \]
\[ M \parallel N \quad \text{parallel connection} \]
\[ \text{let } X = M \text{ in } N \quad \text{let-binding of hole } X \]
\[ \text{bind } (N, \langle a, b \rangle) \quad \text{bind head}(a) \text{ to tail}(b), \text{ where} \]
\[ \langle a, b \rangle \in \text{out}(N) \times \text{in}(N) \]
**Syntax:** formal specification of flow networks

\[ \mathcal{A}, \mathcal{B} \in \text{SMALLNETWORK} \]
\[ X, Y \in \text{HOLE} \]
\[ \mathcal{M}, \mathcal{N} \in \text{NETWORK} \quad ::= \quad \mathcal{A} \quad \text{small network} \]
\[ \quad \mid X \quad \text{hole} \]
\[ \quad \mid \mathcal{M} \parallel \mathcal{N} \quad \text{parallel connection} \]
\[ \quad \mid \text{let } X = \mathcal{M} \text{ in } \mathcal{N} \quad \text{let-binding of hole } X \]
\[ \quad \mid \text{bind } (\mathcal{N}, \langle a, b \rangle) \quad \text{bind } head(a) \text{ to } tail(b), \text{ where } \]
\[ \quad \langle a, b \rangle \in \text{out}(\mathcal{N}) \times \text{in}(\mathcal{N}) \]

**Conditions for well-formedness in full report:**

- Unique arc-naming condition
- Matching-dimensions condition (on input and output arcs)
Syntax: formal specification of flow networks

\[ A, B \in \text{SMALLNETWORK} \]
\[ X, Y \in \text{HOLE} \]
\[ M, N \in \text{NETWORK} \quad ::= \quad A \quad \text{small network} \]
\[ | \quad X \quad \text{hole} \]
\[ | \quad M \parallel N \quad \text{parallel connection} \]
\[ | \quad \text{let } X = M \text{ in } N \quad \text{let-binding of hole } X \]
\[ | \quad \text{bind } (N, \langle a, b \rangle) \quad \text{bind } head(a) \text{ to } tail(b), \text{ where } \langle a, b \rangle \in \text{out}(N) \times \text{in}(N) \]

Conditions for well-formedness in full report:

- Unique arc-naming condition
- Matching-dimensions condition (on input and output arcs)

Derived constructors in full report

(not all derivable from formal syntax above)
Semantics: two basic requirements on flows $f : \mathbb{A} \rightarrow \mathbb{R}^+$
Semantics: two basic requirements on flows $f : A \rightarrow \mathbb{R}^+$

1. Flow conservation at every node:

$$f(a_1) + f(a_2) = f(a_3) + f(a_4)$$
Semantics: two basic requirements on flows \( f : A \rightarrow \mathbb{R}^+ \)

1. Flow conservation at every node:
\[
f(a_1) + f(a_2) = f(a_3) + f(a_4)
\]

2. Capacity constraints at every arc:
\[
L(a) \leq f(a) \leq U(a)
\]
Semantics: two basic requirements on flows $f : A \rightarrow \mathbb{R}^+$

1. Flow conservation at every node:

$$f(a_1) + f(a_2) = f(a_3) + f(a_4)$$

2. Capacity constraints at every arc:

$$L(a) \leq f(a) \leq U(a)$$

Flow $f : A \rightarrow \mathbb{R}^+$ is **feasible** if $f$ satisfies **flow conservation** at every node and **capacity constraints** at every arc.
Semantics: two basic requirements on flows $f : A \rightarrow \mathbb{R}^+$

1. Flow conservation at every node:

$$f(a_1) + f(a_2) = f(a_3) + f(a_4)$$

2. Capacity constraints at every arc:

$$L(a) \leq f(a) \leq U(a)$$

Flow $f : A \rightarrow \mathbb{R}^+$ is **feasible** if $f$ satisfies flow conservation at every node and capacity constraints at every arc.

The **semantics** of flow network $\mathcal{N}$: $[\mathcal{N}] = \{\text{all feasible flows in } \mathcal{N}\}$
**Semantics:** small flow networks with arc capacities

shown: only the non-trivial upper-bound capacities,

omitted: all the trivial upper-bound capacities (= “very large number”) and all the lower-bound thresholds (= 0).
Semantics: generalizations of our framework
**Semantics**: generalizations of our framework

- *Simulating a producer node of 5 units, by adding an input arc:*
Semantics: generalizations of our framework

- *simulating a producer node of 5 units, by adding an input arc:*

- *simulating a consumer node of 3 units, by adding an output arc:*
Semantics: generalizations of our framework

- simulating a **producer** node of 5 units, by adding an **input** arc:

- simulating a **consumer** node of 3 units, by adding an **output** arc:

- analysis is readily extended to multi-commodity flow networks
Semantics: beyond the two basic requirements

Beyond flow conservation at nodes and capacity constraints at arcs, we are also interested in feasible flows satisfying an objective function:
**Semantics: beyond the two basic requirements**

*Beyond flow conservation at nodes and capacity constraints at arcs, we are also interested in feasible flows satisfying an **objective function**:* 

**Minimize Hop Routing (HR)** A minimum hop route is a route with minimal number of links.

**Minimize Arc Utilization (AU)** The utilization of an arc $a$ is defined as  
$$u(a) = \frac{f(a)}{U(a)}.$$ 

**Minimize Mean Delay (MD)** The mean delay of an arc $a$ can be measured by  
$$d(a) = \frac{1}{U(a) - f(a)}.$$  

**Several others**  

...
Syntax-directed **denotational** semantics:

1. If $M = A$, then $\llbracket M \rrbracket = \ldots$
2. If $M = iX$, then $\llbracket M \rrbracket = \ldots$
3. If $M = (P_1 \parallel P_2)$, then $\llbracket M \rrbracket = \ldots$
4. If $M = (\text{let } X = P_1 \text{ in } P_2)$, then $\llbracket M \rrbracket = \ldots$
5. If $M = \text{bind } (P, \langle a, b \rangle)$, then $\llbracket M \rrbracket = \ldots$

Alternative: an operational/reduction semantics.

Fact: The denotational and operational semantics can be shown to be equivalent (in a precise sense to be defined).
Syntax-directed **denotational** semantics:

1. If $\mathcal{M} = \mathcal{A}$, then $[\mathcal{M}] = \ldots$
2. If $\mathcal{M} = i X$, then $[\mathcal{M}] = \ldots$
3. If $\mathcal{M} = (\mathcal{P}_1 \parallel \mathcal{P}_2)$, then $[\mathcal{M}] = \ldots$
4. If $\mathcal{M} = (\text{let } X = \mathcal{P}_1 \text{ in } \mathcal{P}_2)$, then $[\mathcal{M}] = \ldots$
5. If $\mathcal{M} = \text{bind} (\mathcal{P}, \langle a, b \rangle)$, then $[\mathcal{M}] = \ldots$

Alternative: an **operational/reduction** semantics.

**Fact:** The **denotational** and **operational** semantics can be shown to be equivalent (in a precise sense to be defined).
Syntax-directed **denotational** semantics:

1. If $\mathcal{M} = \mathcal{A}$, then $\llbracket \mathcal{M} \rrbracket = \llbracket \mathcal{A} \rrbracket$
2. If $\mathcal{M} = \textit{i}X$, then $\llbracket \mathcal{M} \rrbracket = \textit{i} \llbracket X \rrbracket$
3. If $\mathcal{M} = (\mathcal{P}_1 \parallel \mathcal{P}_2)$, then $\llbracket \mathcal{M} \rrbracket = \{ (f_1 \parallel f_2) | f_i \in \llbracket \mathcal{P}_i \rrbracket \text{ for } i = 1, 2 \}$
4. If $\mathcal{M} = (\textit{let } X = \mathcal{P}_1 \text{ in } \mathcal{P}_2)$, then $\llbracket \mathcal{M} \rrbracket = \llbracket \mathcal{P}_2 \rrbracket$, provided:
   - $\text{dim}(X) \approx \text{dim}(\mathcal{P}_1)$
   - $\llbracket X \rrbracket \approx \{ \llbracket g \rrbracket_A | g \in \llbracket \mathcal{P}_1 \rrbracket \}$ where $A = \text{in}(\mathcal{P}_1) \cup \text{out}(\mathcal{P}_1)$
5. If $\mathcal{M} = \textbf{bind} (\mathcal{P}, \langle a, b \rangle)$, then $\llbracket \mathcal{M} \rrbracket = \{ f | f \in \llbracket \mathcal{P} \rrbracket \text{ and } f(a) = f(b) \}$

Alternative: an **operational/reduction** semantics.

**Fact:** The **denotational** and **operational** semantics can be shown to be equivalent (in a precise sense to be defined).
Types and Typings

- let $\mathcal{N}$ be a flow network, with $A_{\text{in}} = \text{in}(\mathcal{N})$ and $A_{\text{out}} = \text{out}(\mathcal{N})$
Types and Typings

- let $\mathcal{N}$ be a flow network, with $A_{\text{in}} = \text{in}(\mathcal{N})$ and $A_{\text{out}} = \text{out}(\mathcal{N})$

- a **typing** $T$ over $A_{\text{in}} \cup A_{\text{out}}$ is a function:

  $$T : \mathcal{P}^+(A_{\text{in}} \cup A_{\text{out}}) \rightarrow \mathbb{R} \times \mathbb{R}$$

  for every $A \in \mathcal{P}^+(A_{\text{in}} \cup A_{\text{out}})$, typing $T$ assigns an **interval/type** $[r, r']$, possibly empty, of reals.
Types and Typings

- let $\mathcal{N}$ be a flow network, with $A_{\text{in}} = \text{in}(\mathcal{N})$ and $A_{\text{out}} = \text{out}(\mathcal{N})$

- a **typing** $T$ over $A_{\text{in}} \cup A_{\text{out}}$ is a function:

  \[
  T : \mathcal{P}^+(A_{\text{in}} \cup A_{\text{out}}) \rightarrow \mathbb{R} \times \mathbb{R}
  \]

  for every $A \in \mathcal{P}^+(A_{\text{in}} \cup A_{\text{out}})$, typing $T$ assigns an **interval/type** $[r, r']$, possibly empty, of reals
- let $\mathcal{N}$ be a flow network, with $A_{in} = \text{in}(\mathcal{N})$ and $A_{out} = \text{out}(\mathcal{N})$

- a typing $T$ over $A_{in} \cup A_{out}$ is a function:

$$T : \mathcal{P}^+(A_{in} \cup A_{out}) \rightarrow \mathbb{R} \times \mathbb{R}$$

for every $A \in \mathcal{P}^+(A_{in} \cup A_{out})$, typing $T$ assigns an interval/type $[r, r']$, possibly empty, of reals

- $T$ is satisfied by $f : A_{in} \cup A_{out} \rightarrow \mathbb{R}^+$ in $\mathcal{N}$ if and only if

$$r \leq \sum f(A \cap A_{in}) - \sum f(A \cap A_{out}) \leq r'$$
Types and Typings: valid and principal typings

a typing $\mathcal{N} : T$ is \textit{valid} iff:

(sound) every $f : A_{\text{in}} \cup A_{\text{out}} \rightarrow \mathbb{R}^+$ satisfying $T$

can be extended to a feasible flow $g$ in $\mathcal{N}$
Types and Typings: valid and principal typings

a typing $\mathcal{N} : T$ is valid iff:

(sound) every $f : A_{in} \cup A_{out} \rightarrow \mathbb{R}^+$ satisfying $T$

   can be extended to a feasible flow $g$ in $\mathcal{N}$

a typing $\mathcal{N} : T$ is principal iff $\mathcal{N} : T$ is valid and:

(complete) every feasible flow $g$ in $\mathcal{N}$ satisfies $T$

   (i.e., the restriction of $g$ to $A_{in} \cup A_{out}$ satisfies $T$)
Types and Typings: valid and principal typings

A typing $\mathcal{N} : T$ is **valid** iff:

- (sound) every $f : \mathcal{A}_{\text{in}} \cup \mathcal{A}_{\text{out}} \rightarrow \mathbb{R}^+$ satisfying $T$
  - can be extended to a feasible flow $g$ in $\mathcal{N}$

A typing $\mathcal{N} : T$ is **principal** iff $\mathcal{N} : T$ is valid **and**:

- (complete) every feasible flow $g$ in $\mathcal{N}$ satisfies $T$
  - (i.e., the restriction of $g$ to $\mathcal{A}_{\text{in}} \cup \mathcal{A}_{\text{out}}$ satisfies $T$)
Example: a typing for small network $\mathcal{A}$

\[
\begin{align*}
\{a_1\} &: [0, 15] & \{a_2\} &: [0, 25] & \{a_3\} &: [-15, 0] & \{a_4\} &: [-25, 0] \\
\{a_1, a_2\} &: [0, 30] & \{a_1, a_3\} &: [-10, 10] & \{a_1, a_4\} &: [-25, 15] \\
\{a_2, a_3\} &: [-15, 25] & \{a_2, a_4\} &: [-10, 10] & \{a_3, a_4\} &: [-30, 0] \\
\{a_1, a_2, a_3\} &: [0, 25] & \{a_1, a_2, a_4\} &: [0, 15] \\
\{a_1, a_3, a_4\} &: [-25, 0] & \{a_2, a_3, a_4\} &: [-15, 0] \\
\{a_1, a_2, a_3, a_4\} &: [0, 0]
\end{align*}
\]
Example: a typing for small network $B$

\[
\begin{align*}
\{b_1\} & : [0, 15] & \{b_2\} & : [0, 25] & \{b_3\} & : [-15, 0] & \{b_4\} & : [-25, 0] \\
\{b_1, b_2\} & : [0, 30] & \{b_1, b_3\} & : [-10, 12] & \{b_1, b_4\} & : [-25, 15] \\
\{b_2, b_3\} & : [-15, 25] & \{b_2, b_4\} & : [-12, 10] & \{b_3, b_4\} & : [-30, 0] \\
\{b_1, b_2, b_3\} & : [0, 25] & \{b_1, b_2, b_4\} & : [0, 15] \\
\{b_1, b_3, b_4\} & : [-25, 0] & \{b_2, b_3, b_4\} & : [-15, 0] \\
\{b_1, b_2, b_3, b_4\} & : [0, 0]
\end{align*}
\]
Example: the typing for small network $\mathcal{A}$ is principal.

\[
\begin{align*}
\{a_1\} &: [0, 15] \\
\{a_2\} &: [0, 25] \\
\{a_3\} &: [-15, 0] \\
\{a_4\} &: [-25, 0] \\
\{a_1, a_2\} &: [0, 30] \\
\{a_1, a_3\} &: [-10, 10] \\
\{a_1, a_4\} &: [-25, 15] \\
\{a_2, a_3\} &: [-15, 25] \\
\{a_2, a_4\} &: [-10, 10] \\
\{a_3, a_4\} &: [-30, 0] \\
\{a_1, a_2, a_3\} &: [0, 25] \\
\{a_1, a_2, a_4\} &: [0, 15] \\
\{a_1, a_3, a_4\} &: [-25, 0] \\
\{a_2, a_3, a_4\} &: [-15, 0] \\
\{a_1, a_2, a_3, a_4\} &: [0, 0]
\end{align*}
\]
Example: the typing for small network $\mathcal{B}$ is principal

\[
\begin{align*}
\{b_1\} & : [0, 15] & \{b_2\} & : [0, 25] & \{b_3\} & : [-15, 0] & \{b_4\} & : [-25, 0] \\
\{b_1, b_2\} & : [0, 30] & \{b_1, b_3\} & : [-10, 12] & \{b_1, b_4\} & : [-25, 15] \\
\{b_2, b_3\} & : [-15, 25] & \{b_2, b_4\} & : [-12, 10] & \{b_3, b_4\} & : [-30, 0] \\
\{b_1, b_2, b_3\} & : [0, 25] & \{b_1, b_2, b_4\} & : [0, 15] \\
\{b_1, b_3, b_4\} & : [-25, 0] & \{b_2, b_3, b_4\} & : [-15, 0] \\
\{b_1, b_2, b_3, b_4\} & : [0, 0]
\end{align*}
\]
Types and Typings: typing-inference for small flow networks

Lemma

Let $A$ be a small flow network with $A_{in} = \text{in}(A)$ and $A_{out} = \text{out}(A)$. We can compute a typing $T$ over $A_{in} \cup A_{out}$ such that for every $f : A_{in} \cup A_{out} \to \mathbb{R}^+$:

$f$ can be extended to a feasible flow in $A$ $\iff$ $f$ satisfies $T$

Corollary

For every small flow network $A$, we can infer a principal typing $T$ for $A$ back to satisfaction.
Lemma:
Let $\mathcal{A}$ be a small flow network with $A_{\text{in}} = \text{in}(\mathcal{A})$ and $A_{\text{out}} = \text{out}(\mathcal{A})$. We can compute a typing $T$ over $A_{\text{in}} \cup A_{\text{out}}$ such that for every $f : A_{\text{in}} \cup A_{\text{out}} \to \mathbb{R}^+$:

$$f \text{ can be extended to a feasible flow in } \mathcal{A} \iff f \text{ satisfies } T$$
Types and Typings: typing-inference for small flow networks

**Lemma:**
Let $\mathcal{A}$ be a small flow network with $A_{in} = \text{in}(\mathcal{A})$ and $A_{out} = \text{out}(\mathcal{A})$. We can compute a typing $T$ over $A_{in} \cup A_{out}$ such that for every $f : A_{in} \cup A_{out} \rightarrow \mathbb{R}^+$:

$$f \text{ can be extended to a feasible flow in } \mathcal{A} \iff f \text{ satisfies } T$$

**Corollary:**
For every small flow network $\mathcal{A}$, we can infer a principal typing $T$ for $\mathcal{A}$
Typing Rules

\[
\text{HOLE} \quad \frac{(X : T) \in \Gamma}{\Gamma \vdash iX : iT}
\]

available renaming index \( i \geq 1 \)
Typing Rules

HOLE

\[ \frac{(X : T) \in \Gamma}{\Gamma \vdash iX : iT} \]

available renaming index \( i \geq 1 \)

SMALL

PAR

BIND

LET
Typing Rules

**HOLE**

\[
\frac{(X : T) \in \Gamma}{\Gamma \vdash iX : iT}
\]

available renaming index \(i \geq 1\)

**SMALL**

\[
\frac{}{\Gamma \vdash A : T}
\]

typing \(T\) for small network \(A\)

**PAR**

\[
\frac{\Gamma \vdash \mathcal{N}_1 : T_1 \quad \Gamma \vdash \mathcal{N}_2 : T_2}{\Gamma \vdash (\mathcal{N}_1 \parallel \mathcal{N}_2) : (T_1 \parallel T_2)}
\]

**BIND**

\[
\frac{\Gamma \vdash \mathcal{N} : T}{\Gamma \vdash \text{bind}(\mathcal{N}, \langle a, b \rangle) : \text{bind}(T, \langle a, b \rangle)}
\]

\(\langle a, b \rangle \in \text{out}(\mathcal{N}) \times \text{in}(\mathcal{N})\)

**LET**

\[
\frac{}{\Gamma \vdash M : T_1 \cup \{(X : T)\}}
\]

\[
\frac{}{\Gamma \vdash \text{let } X = M \text{ in } \mathcal{N} : T_2}
\]

\(T_1 \approx T_2\)
Typing Rules

HOLE
\[ \frac{(X : T) \in \Gamma}{\Gamma \vdash iX : iT} \]
available renaming index \(i \geq 1\)

SMALL
\[ \frac{}{\Gamma \vdash A : T} \]
typing \(T\) for small network \(A\)

PAR
\[ \frac{\Gamma \vdash N_1 : T_1 \quad \Gamma \vdash N_2 : T_2}{\Gamma \vdash (N_1 \parallel N_2) : (T_1 \parallel T_2)} \]

BIND
\[ \frac{}{\Gamma \vdash \text{bind}(N, \langle a, b \rangle) : \text{bind}(T, \langle a, b \rangle)} \]
\(\langle a, b \rangle \in \text{out}(N) \times \text{in}(N)\)

LET
\[ \frac{\Gamma \vdash M : T_1 \quad \Gamma \cup \{ (X : T_2) \} \vdash N : T}{\Gamma \vdash (\text{let } X = M \text{ in } N) : T} \]
\(T_1 \approx T_2\)
Typing Rules

**Theorem:**

Let $\mathcal{N}$ be a closed flow network specification and $T$ a typing for $\mathcal{N}$ derived according to the rules of the typing system, i.e., the judgment “$\vdash \mathcal{N} : T$” is formally derivable.

If the typing of every small network $\mathcal{A}$ in $\mathcal{N}$ is valid (principal) for $\mathcal{A}$, then $T$ is a valid (principal) typing for $\mathcal{N}$.
Relativized Semantics: some objective functions

Minimize Hop Routing (HR) A minimum hop route is a route with minimal number of links.

Minimize Arc Utilization (AU) The utilization of an arc $a$ is defined as $u_a = f_a / \sum_U(a)$.

Minimize Mean Delay (MD) The mean delay of an arc $a$ can be measured by $d_a = 1 / (\sum_U(a) - f_a)$.

Several others...

All of the form $\alpha : JN\rightarrow R$. $\alpha$ assigns a real value $r$ to every feasible flow $f$ in $N$, preferred feasible flows $f$ minimize (or maximize) $\alpha(f)$ subject to the (mild) condition $\alpha(f) = \sum_{a \in A} \theta(f(a)) / \sum_{a \in A}$ for some appropriate $\theta$, one for each $\alpha$.
Relativized Semantics: some objective functions

**Minimize Hop Routing (HR)** A minimum hop route is a route with minimal number of links.

**Minimize Arc Utilization (AU)** The utilization of an arc $a$ is defined as

$$u(a) = f(a)/U(a).$$

**Minimize Mean Delay (MD)** The mean delay of an arc $a$ can be measured by

$$d(a) = 1/(U(a) - f(a)).$$

Several others . . .
Relativized Semantics: some objective functions

**Minimize Hop Routing (HR)** A minimum hop route is a route with minimal number of links.

**Minimize Arc Utilization (AU)** The utilization of an arc \( a \) is defined as
\[
u(a) = \frac{f(a)}{U(a)}.
\]

**Minimize Mean Delay (MD)** The mean delay of an arc \( a \) can be measured by
\[
d(a) = \frac{1}{(U(a) - f(a))}.
\]

Several others . . .

**All of the form** \( \alpha : [\mathcal{N}] \rightarrow \mathbb{R}^+ \)
\[\alpha \text{ assigns a real value } r \text{ to every feasible flow } f \text{ in } \mathcal{N}, \]
preferred feasible flows \( f \text{ minimize (or maximize) } \alpha(f). \)
Relativized Semantics: some objective functions

**Minimize Hop Routing (HR)** A minimum hop route is a route with minimal number of links.

**Minimize Arc Utilization (AU)** The utilization of an arc $a$ is defined as 

$$u(a) = \frac{f(a)}{U(a)}.$$  

**Minimize Mean Delay (MD)** The mean delay of an arc $a$ can be measured by 

$$d(a) = \frac{1}{U(a) - f(a)}.$$  

Several others . . .

All of the form $\alpha : [\mathcal{N}] \to \mathbb{R}^+$ 

$\alpha$ assigns a real value $r$ to every feasible flow $f$ in $\mathcal{N}$, preferred feasible flows $f$ **minimize** (or **maximize**) $\alpha(f)$.

satisfying the (mild) condition $\alpha(f) = \sum \{ \theta(f(a)) \mid a \in A_\# \cup A_{out} \}$ 

for some appropriate $\theta$, one for each $\alpha$.

skip elaboration
Relativized Semantics: additive aggregate functions

- The objective $\alpha : [\mathcal{N}] \to \mathbb{R}^+$ is an **additive aggregate function** if there is a function $\theta : \mathbb{R}^+ \to \mathbb{R}^+$ such that

$$\alpha(f) = \sum \{ \theta(f(a)) \mid a \in A_\# \cup A_{\text{out}} \}$$
Relativized Semantics: additive aggregate functions

- The objective \( \alpha : [\mathcal{N}] \rightarrow \mathbb{R}^+ \) is an **additive aggregate function** if there is a function \( \theta : \mathbb{R}^+ \rightarrow \mathbb{R}^+ \) such that

\[
\alpha(f) = \sum \{ \theta(f(a)) \mid a \in A_{\#} \cup A_{\text{out}} \}
\]

- Some objective functions that are additive aggregate:
  - minimize hop routing \( \text{HR} : [\mathcal{N}] \rightarrow \mathbb{R}^+ \),
  - minimize arc utilization \( \text{AU} : [\mathcal{N}] \rightarrow \mathbb{R}^+ \),
  - minimize mean delay \( \text{MD} : [\mathcal{N}] \rightarrow \mathbb{R}^+ \),
  - and many others . . .

But not all objective functions are. For example, \( \alpha_r(f) = \text{if} \sum f(A_{\text{in}}) \text{then} \text{HR}(f) \text{else} \text{AU}(f) \) is not additive aggregate, where \( \alpha_r \) is a fixed number.
The objective $\alpha : [\mathcal{N}] \rightarrow \mathbb{R}^+$ is an **additive aggregate function** if there is a function $\theta : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ such that

$$\alpha(f) = \sum \{ \theta(f(a)) | a \in A_{\#} \cup A_{\text{out}} \}$$

Some objective functions that are additive aggregate:
- minimize hop routing $HR : [\mathcal{N}] \rightarrow \mathbb{R}^+$,
- minimize arc utilization $AU : [\mathcal{N}] \rightarrow \mathbb{R}^+$,
- minimize mean delay $MD : [\mathcal{N}] \rightarrow \mathbb{R}^+$,
- and many others . . .

But **not** all objective functions are. For example,

$$\alpha_r(f) = \text{if } \sum f(A_{\text{in}}) \leq r \text{ then } HR(f) \text{ else } AU(f)$$

is **not** additive aggregate, where $r$ is a fixed number.
Relativized Semantics

The **semantics** of flow network $\mathcal{N}$ relative to objective function $\alpha$:

$$[[\mathcal{N} \mid \alpha]] = \{ \text{all feasible flows in } \mathcal{N} \text{ satisfying } \alpha \}$$
The **semantics** of flow network $\mathcal{N}$ relative to objective function $\alpha$:

$$\llbracket \mathcal{N} \mid \alpha \rrbracket = \{ \text{all feasible flows in } \mathcal{N} \text{ satisfying } \alpha \}$$

$$= \{ f \in \llbracket \mathcal{N} \rrbracket \mid f \text{ satisfies } \alpha \}$$
Relativized Semantics

The **semantics** of flow network $\mathcal{N}$ relative to objective function $\alpha$:

$$[[\mathcal{N}|\alpha]] = \{ \text{all feasible flows in } \mathcal{N} \text{ satisfying } \alpha \}$$

$$= \{ f \in [[\mathcal{N}]] | f \text{ satisfies } \alpha \}$$
Relativized Semantics

The **semantics** of flow network $\mathcal{N}$ relative to objective function $\alpha$:

$$[[\mathcal{N}] | \alpha] = \{ \text{all feasible flows in } \mathcal{N} \text{ satisfying } \alpha \}$$

$$= \{ f \in [[\mathcal{N}]] \mid f \text{ satisfies } \alpha \}$$

$$= \{ \langle f, B, r \rangle \mid f \in [[\mathcal{N}]], \ B \subseteq \text{in}(\mathcal{N}) \cup \text{out}(\mathcal{N}), \ r = \alpha(f), \ and \ for \ every \ g \in [[\mathcal{N}]], \ if \ \lfloor f \rfloor_B = \lfloor g \rfloor_B \ then \ \alpha(f) \leq \alpha(g) \}$$
Relativized Semantics: formal definition

Syntax-directed **denotational** semantics:

1. If $M = A$, then $\llbracket M | \alpha \rrbracket = \cdots$
2. If $M = i X$, then $\llbracket M | \alpha \rrbracket = \cdots$
3. If $M = (P_1 \parallel P_2)$, then $\llbracket M | \alpha \rrbracket = \cdots$

4. If $M = (\text{let } X = P \text{ in } P')$, then $\llbracket M | \alpha \rrbracket = \cdots$

5. If $M = \text{bind} (P, \langle a, b \rangle)$, then $\llbracket M | \alpha \rrbracket = \cdots$
Relativized Semantics: formal definition

Syntax-directed **denotational** semantics:

1. If $M = A$, then $\llbracket M \mid \alpha \rrbracket = \llbracket A \mid \alpha \rrbracket$

2. If $M = iX$, then $\llbracket M \mid \alpha \rrbracket = i\llbracket X \mid \alpha \rrbracket$

3. If $M = (P_1 \parallel P_2)$, then $\llbracket M \mid \alpha \rrbracket =$

   $\{ \langle f_1 \parallel f_2, B_1 \cup B_2, r_1 + r_2 \rangle \mid \langle f_i, B_i, r_i \rangle \in \llbracket P_i \mid \alpha \rrbracket \text{ for } i = 1, 2 \}$

4. If $M = \left( \text{let } X = P \text{ in } P' \right)$, then $\llbracket M \mid \alpha \rrbracket = \llbracket P' \mid \alpha \rrbracket$, provided:
   
   - $\dim(X) \approx \dim(P)$,
   
   - $\llbracket X \mid \alpha \rrbracket \approx \{ \langle [g]_A, C, r \rangle \mid \langle g, C, r \rangle \in \llbracket P \mid \alpha \rrbracket, A = \text{in}(P) \cup \text{out}(P) \}$

5. If $M = \text{bind}(P, \langle a, b \rangle)$, then $\llbracket M \mid \alpha \rrbracket =$

   $\{ \langle f, B, r \rangle \mid \langle f, B \cup \{ a, b \}, r \rangle \in \llbracket P \mid \alpha \rrbracket, f(a) = f(b),$

   and for every $\langle g, B \cup \{ a, b \}, s \rangle \in \llbracket P \mid \alpha \rrbracket$

   if $g(a) = g(b)$ and $\llbracket f \rrbracket_B = \llbracket g \rrbracket_B$ then $r \leq s \}$
**HOLE**  
\[
\frac{(X : (T, \Phi)) \in \Gamma}{\Gamma \vdash iX : (iT, i\Phi)}
\]

\(i \geq 1\) is smallest available renaming index
Relativized Typings

**HOLE**

\[
\frac{(X : (T, \Phi)) \in \Gamma}{\Gamma \vdash iX : (iT, i\Phi)}
\]

\(i \geq 1\) is smallest available renaming index

**SMALL**

\[
\frac{}{\Gamma \vdash A : (T, \Phi)}
\]

\((T, \Phi)\) is a relativized typing for small network \(A\)
Relativized Typings

\[
\begin{aligned}
\text{HOLE} & \quad \frac{(X : (T, \Phi)) \in \Gamma}{\Gamma \vdash i X : (i T, i \Phi)} \\
\text{SMALL} & \quad \frac{}{\Gamma \vdash A : (T, \Phi)} \\
\text{PAR} & \\
\text{BIND} & \\
\text{LET} & 
\end{aligned}
\]

\(i \geq 1\) is smallest available renaming index

\((T, \Phi)\) is a relativized typing for small network \(A\)

DSL for Flow Network Design  Syntax  Semantics  Typings  Rules  Rel Semantics  Rel Typings  End
HOLE

\[
\frac{(X : (T, \Phi)) \in \Gamma}{\Gamma \vdash iX : (iT, i\Phi)}
\]
i \geq 1 \text{ is smallest available renaming index}

SMALL

\[
\Gamma \vdash A : (T, \Phi)
\]

(T, \Phi) is a relativized typing for small network A

PAR

\[
\frac{\Gamma \vdash N_1 : (T_1, \Phi_1) \quad \Gamma \vdash N_2 : (T_2, \Phi_2)}{\Gamma \vdash (N_1 \parallel N_2) : (T_1, \Phi_1) \parallel (T_2, \Phi_2)}
\]

BIND

LET
Relativized Typings

**HOLE**

\[
\frac{(X : (T, \Phi)) \in \Gamma}{\Gamma \vdash \ iX : (iT, \ i\Phi)}
\]

*i* ≥ 1 is smallest available renaming index

**SMALL**

\[
\frac{}{\Gamma \vdash \ A : (T, \Phi)}
\]

(\(T, \Phi\)) is a relativized typing for small network \(A\)

**PAR**

\[
\frac{\Gamma \vdash \ N_1 : (T_1, \Phi_1) \quad \Gamma \vdash \ N_2 : (T_2, \Phi_2)}{\Gamma \vdash \ (N_1 \parallel N_2) : (T_1, \Phi_1) \parallel (T_2, \Phi_2)}
\]

**BIND**

\[
\frac{\Gamma \vdash \ N : (T, \Phi)}{\Gamma \vdash \ \text{bind}(\ N, \langle a, b \rangle) : \text{bind}(\ (T, \Phi), \langle a, b \rangle)}
\]

\(\langle a, b \rangle \in \text{out}(N) \times \text{in}(N)\)

**LET**

**skip elaboration**
Relativized Typings

**HOLE**

\[
\frac{(X : (T, \Phi)) \in \Gamma}{\Gamma \vdash iX : (iT, i\Phi)}
\]

\(i \geq 1\) is smallest available renaming index

**SMALL**

\[
\frac{\Gamma \vdash A : (T, \Phi)}{}
\]

\((T, \Phi)\) is a relativized typing for small network \(A\)

**PAR**

\[
\frac{\Gamma \vdash N_1 : (T_1, \Phi_1) \quad \Gamma \vdash N_2 : (T_2, \Phi_2)}{\Gamma \vdash (N_1 \parallel N_2) : (T_1, \Phi_1) \parallel (T_2, \Phi_2)}
\]

**BIND**

\[
\frac{\Gamma \vdash N : (T, \Phi)}{\Gamma \vdash \text{bind}(N, \langle a, b \rangle) : \text{bind}((T, \Phi), \langle a, b \rangle)}
\]

\(\langle a, b \rangle \in \text{out}(N) \times \text{in}(N)\)

**LET**

\[
\frac{\Gamma \vdash M : (T_1, \Phi_1) \quad \Gamma \cup \{X : (T_2, \Phi_2)\} \vdash N : (T, \Phi)}{\Gamma \vdash (\text{let } X = M \text{ in } N) : (T, \Phi)}
\]

\((T_1, \Phi_1) \approx (T_2, \Phi_2)\)
Thank You!
Related Presentations

1. “A Domain-Specific Language for Incremental and Modular Design of Large-Scale Verifiably-Safe Flow Networks” (with A. Bestavros), in IFIP DSL 2011, Sept 2011.


... and thanks to my collaborators in the iBench Initiative:

Azer Bestavros, Andrei Lapets, Vatche Ishakian,
Michael Ocean, Nate Soule.

https://sites.google.com/site/ibenchbu/