Lightweight Formal Methods for the Development of High-Assurance Network Systems

Assaf Kfoury

with contributions from

Azer Bestavros, Adam Bradley, Andrei Lapets, and Michael Ocean

iBench Initiative
http://www.cs.bu.edu/groups/ibench/
snBench
http://csr.bu.edu/snbench/

Computer Science

Specification of an unchecked traffic module

$$\mathcal{A} \quad = \quad \boxed{(N, E, I, O, \mathsf{PP}, \mathsf{Con})}$$

Specification of an unchecked traffic module

$$\mathcal{A} = (N, E, I, O, \mathsf{PP}, \mathsf{Con})$$

N = finite set of traffic junctions (nodes) in $\mathcal{A}$

$E \subseteq N \times N$, finite set of internal links (edges) in $\mathcal{A}$

I = finite ordered multi-set of entering (input) links, each of the form $\langle \_, n \rangle$ for some $n \in N$

O = finite ordered multi-set of exiting (output) links, each of the form $\langle n, \_ \rangle$ for some $n \in N$

PP = injective map from $E \cup I \cup O$ to $(velocity \cdot density)$ pairs of parameters $\{v_0 \cdot k_0, v_1 \cdot k_1, v_2 \cdot k_2, \ldots\}$

Con = finite set of constraints over $\mathsf{parameters}(E \cup I \cup O)$

Specification of a checked (type-safe) traffic module

$$\mathcal{A} \; : \; \mathsf{Typ} \,|\, \mathsf{Con}^{*} \quad = \quad (N, E, I, O, \mathsf{PP}, \mathsf{Con}) \; : \; \mathsf{Typ} \,|\, \mathsf{Con}^{*}$$

Specification of a checked (type-safe) traffic module

$$\mathcal{A} \;:\; \mathsf{Typ}\,|\,\mathsf{Con}^* \quad = \quad (N, E, I, O, \mathsf{PP}, \mathsf{Con}) \;:\; \mathsf{Typ}\,|\,\mathsf{Con}^*$$

where

$(N, E, I, O, \mathsf{PP}, \mathsf{Con})$ — **given traffic module**

$\mathsf{Typ}\,|\,\mathsf{Con}^*$ — **inferred types and constraints**

Specification of a checked (type-safe) traffic module

$$\mathcal{A} \; : \; \mathsf{Typ} \,|\, \mathsf{Con}^* \quad = \quad (N, E, I, O, \mathsf{PP}, \mathsf{Con}) \; : \; \mathsf{Typ} \,|\, \mathsf{Con}^*$$

where

$(N, E, I, O, \mathsf{PP}, \mathsf{Con})$ — **given traffic module**

$\mathsf{Typ} \,|\, \mathsf{Con}^*$ — **inferred types and constraints**

$$\begin{aligned} \mathsf{Typ} \; &= \text{map from } \mathsf{PP}(I \cup O) \text{ to the set of } (\mathit{velocity} \cdot \mathit{density}) \text{ types} \\ &\quad \{[a_1, a_2] \cdot [b_1, b_2] \,|\, a_1, a_2, b_1, b_2 \in \mathbb{N} \text{ and } a_1 \leq a_2, b_1 \leq b_2\} \end{aligned}$$

$\mathsf{Con}^* =$ finite set of $\boldsymbol{\mathit{linear}}$ constraints over $\mathsf{parameters}(E \cup I \cup O)$

such that **condition** $[\mathbf{Pre/Post}]$ is satisfied.

Specification of a checked (type-safe) traffic module

Condition [Pre/Post] For every valuation $\textsc{val} : \textsf{parameters}(E \cup I \cup O) \to \mathbb{N}$

IF	• $\textsc{val} \models \mathsf{Typ}(\mathsf{PP}(I))$	inferred input types
	• $\textsc{val} \models \mathsf{Con}^*$	inferred linear constraints
	• $\textsc{val} \models \mathsf{Con}$	given internal constraints
THEN	• $\textsc{val} \models \mathsf{Typ}(\mathsf{PP}(O))$	inferred output types

In words: *"Every valuation satisfying the inferred input types, the inferred linear constraints, and the given internal constraints, satisfies the inferred output types."*

Specification of a checked (type-safe) traffic module

- We can always define
  Con^*, $\mathsf{Typ}(\mathsf{PP}(I))$, and $\mathsf{Typ}(\mathsf{PP}(0))$, to satisfy [Pre/Post].

- GOAL: Define strongest [Pre/Post], i.e., find *weakest precondition*:

  - least restrictive Con^*
  - least restrictive $\mathsf{Typ}(\mathsf{PP}(I))$

  and *strongest postcondition*:

  - most restrictive $\mathsf{Typ}(\mathsf{PP}(O))$

  such that [Pre/Post] is satisfied.

- COMPLICATION: Strongest [Pre/Post] is not uniquely defined.

From Module Specification to Network Specification

<u>**Modules**</u> $\mathcal{A}, \mathcal{B}, \mathcal{C}, \ldots$

<u>**Networks**</u> $\mathcal{M}, \mathcal{N}, \mathcal{P}, \ldots$
For every network $\mathcal{M}$

$$\mathsf{In}(\mathcal{M}) = \langle i_1, i_2, \ldots \rangle \text{ finite sequence of entering links}$$

$$\mathsf{Out}(\mathcal{M}) = \langle o_1, o_2, \ldots \rangle \text{ finite sequence of exiting links}$$

<u>**Variables**</u> $X, Y, Z, \ldots$ unknown networks, yet to be internally specified
For every variable X

$$\mathsf{In}(X) = \langle i_1, i_2, \ldots \rangle$$

$$\mathsf{Out}(X) = \langle o_1, o_2, \ldots \rangle$$

Γ set of variables, each with its in/out links, written as a list

DSL for Network Specification (untyped)

$$\textsc{Module} \qquad \frac{\mathcal{A}\ \text{Module}}{\Gamma \vdash \mathcal{A} : (I, O)} \qquad I = \mathsf{In}(\mathcal{A}), \quad O = \mathsf{Out}(\mathcal{A})$$

$$\textsc{Variable} \qquad \frac{\{X : (I, O)\} \subseteq \Gamma}{\Gamma \vdash X : (I, O)} \qquad I = \mathsf{In}(X), \quad O = \mathsf{Out}(X)$$

DSL for Network Specification (untyped)

Connect
$$\frac{\Gamma \vdash \mathcal{M} : (I_1, O_1) \qquad \Gamma \vdash \mathcal{N} : (I_2, O_2)}{\Gamma \vdash \textbf{conn}_{\theta,m,n}(\mathcal{M}, \mathcal{N}) : (I, O)}$$

$$\mathsf{parameters}(\mathcal{M}) \cap \mathsf{parameters}(\mathcal{N}) = \varnothing$$

$$m = \mathsf{length}(O_1) \qquad n = \mathsf{length}(I_2)$$

$$\theta : \subseteq_{1\text{-}1} \{1, \ldots, m\} \times \{1, \ldots, n\}$$

$$I = I_1 \cdot (I_2/\mathsf{range}(\theta)) \qquad O = (O_1/\mathsf{domain}(\theta)) \cdot O_2$$

Loop
$$\frac{\Gamma \vdash \mathcal{M} : (I, O)}{\Gamma \vdash \textbf{loop}_{\theta,m,n}(\mathcal{M}) : (I', O')}$$

$$m = \mathsf{length}(O) \qquad n = \mathsf{length}(I)$$

$$\theta : \subseteq_{1\text{-}1} \{1, \ldots, m\} \times \{1, \ldots, n\}$$

$$I' = I/\mathsf{range}(\theta) \qquad O' = O/\mathsf{domain}(\theta)$$

If $A = \langle a_1, a_2, a_3, \ldots \rangle$ and $X \subseteq \{1, 2, 3, \ldots\}$ then $A/X = \langle a_{n_1}, a_{n_2}, a_{n_3}, \ldots \rangle$ where $n_1 < n_2 < n_3 < \ldots$ and $\{n_1, n_2, n_3, \ldots\} = \{1, 2, 3, \ldots\} - X$.

DSL for Network Specification (untyped)

$$\frac{\Gamma \vdash \mathcal{M}_1 : (I_1, O_1) \;\cdots\; \Gamma \vdash \mathcal{M}_n : (I_n, O_n) \qquad \Gamma, X : (I, O) \vdash \mathcal{N} : (I', O')}{\Gamma \vdash \textbf{let } X \in \{\mathcal{M}_1, \ldots, \mathcal{M}_n\} \textbf{ in } \mathcal{N} \; : (I', O')}$$

for every $1 \leq m \leq n$

$\mathsf{parameters}(\mathcal{M}_m) \cap \mathsf{parameters}(\mathcal{N}) = \varnothing$

$\mathsf{length}(I_m) = \mathsf{length}(I) \quad \text{and} \quad \mathsf{length}(O_m) = \mathsf{length}(O)$

Types and Sequences of Typed Links

Types σ, τ

Every type σ is a pair of intervals $\sigma = [a_1, a_2] \cdot [b_1, b_2]$

Sequences of links with types

$\langle \ell_1 : \sigma_1, \ell_2 : \sigma_2, \ldots \rangle$

If $L = \langle \ell_1 : \sigma_1, \ell_2 : \sigma_2, \ldots \rangle$ then $\mathsf{types}(L) = \langle \sigma_1, \sigma_2, \ldots \rangle$.

Subtyping <:

For all types $\sigma = [a_1, a_2] \cdot [b_1, b_2]$ and $\tau = [a_3, a_4] \cdot [b_3, b_4]$

$$\boxed{\boldsymbol{\sigma <: \tau} \quad \text{iff} \quad [a_1, a_2] \subseteq [a_3, a_4] \text{ and } [b_1, b_2] \subseteq [b_3, b_4]}$$

For all sequences of types $S = \langle \sigma_1, \sigma_2, \ldots \rangle$ and $T = \langle \tau_1, \tau_2, \ldots \rangle$

$$\boxed{\boldsymbol{S <: T} \quad \text{iff} \quad \sigma_1 <: \tau_1, \sigma_2 <: \tau_2, \ldots}$$

For all sequences of links with types
$L = \langle \ell_1 : \sigma_1, \ell_2 : \sigma_2, \ldots \rangle$ and $M = \langle \ell_1 : \tau_1, \ell_2 : \tau_2, \ldots \rangle$

$$\boxed{\boldsymbol{L <: M} \quad \text{iff} \quad \mathsf{types}(L) <: \mathsf{types}(M)}$$

DSL for Network Specification (with types, no constraints)

Same notational conventions as for the untyped DSL

Networks

For every network $\mathcal{M}$

$$\mathsf{In}(\mathcal{M}) = \langle i_1 : \sigma_1,\ i_2 : \sigma_2,\ \ldots\rangle \quad \text{finite sequence of typed links}$$

$$\mathsf{Out}(\mathcal{M}) = \langle o_1 : \tau_1,\ o_2 : \tau_2,\ \ldots\rangle \quad \text{finite sequence of typed links}$$

Variables

For every variable X

$$\mathsf{In}(X) = \langle i_1 : \sigma_1,\ i_2 : \sigma_2,\ \ldots\rangle$$

$$\mathsf{Out}(X) = \langle o_1 : \tau_1,\ o_2 : \tau_2,\ \ldots\rangle$$

No change in the satement of rules MODULE and VARIABLE.

DSL for Network Specification (with types, no constraints)

$$\textsc{Connect}\qquad \frac{\Gamma \vdash \mathcal{M} : (I_1, O_1) \qquad \Gamma \vdash \mathcal{N} : (I_2, O_2)}{\Gamma \vdash \textbf{conn}_{\theta,m,n}(\mathcal{M}, \mathcal{N}) : (I, O)}$$

$$\mathsf{parameters}(\mathcal{M}) \cap \mathsf{parameters}(\mathcal{N}) = \varnothing$$

$$m = \mathsf{length}(O_1) \qquad n = \mathsf{length}(I_2)$$

$$\theta : \subseteq_{\text{1-1}} \{1, \ldots, m\} \times \{1, \ldots, n\}$$

$$I = I_1 \cdot (I_2/\mathsf{range}(\theta)) \qquad O = (O_1/\mathsf{domain}(\theta)) \cdot O_2$$

$$\boxed{[\mathsf{types}(O_1)]_p <: [\mathsf{types}(I_2)]_q \quad \text{for every } (p, q) \in \theta}$$

If p-th outgoing link from $\mathcal{M}$ is connected to q-th incoming link to $\mathcal{N}$ then *the outgoing-link type is a subtype of the incoming-link type.*

Rule LOOP adjusted in same way as rule CONNECT.

DSL for Network Specification (with types, no constraints)

$$\textsc{Let}\quad \frac{\Gamma \vdash \mathcal{M}_1 : (I_1, O_1) \ \cdots \ \Gamma \vdash \mathcal{M}_n : (I_n, O_n) \qquad \Gamma, X : (I, O) \vdash \mathcal{N} : (I', O')}{\Gamma \vdash \textbf{let } X \in \{\mathcal{M}_1, \ldots, \mathcal{M}_n\} \textbf{ in } \mathcal{N} \ : (I', O')}$$

for every $1 \leq m \leq n$

$\mathsf{parameters}(\mathcal{M}_m) \cap \mathsf{parameters}(\mathcal{N}) = \varnothing$

$\mathsf{length}(I_m) = \mathsf{length}(I)$ and $\mathsf{length}(O_m) = \mathsf{length}(O)$

$$\boxed{\begin{array}{l} \mathsf{types}(I_1) = \cdots = \mathsf{types}(I_n) = \mathsf{types}(I) \ \text{ and} \\ \mathsf{types}(O_1) = \cdots = \mathsf{types}(O_n) = \mathsf{types}(O) \end{array}}$$

DSL for Network Specification (with types, no constraints)

$$\textsc{Let}\quad \frac{\Gamma \vdash \mathcal{M}_1 : (I_1, O_1) \quad \cdots \quad \Gamma \vdash \mathcal{M}_n : (I_n, O_n) \qquad \Gamma, X : (I, O) \vdash \mathcal{N} : (I', O')}{\Gamma \vdash \textbf{let } X \in \{\mathcal{M}_1, \ldots, \mathcal{M}_n\} \textbf{ in } \mathcal{N} \ : (I', O')}$$

for every $1 \leq m \leq n$

$\mathsf{parameters}(\mathcal{M}_m) \cap \mathsf{parameters}(\mathcal{N}) = \varnothing$

$\mathsf{length}(I_m) = \mathsf{length}(I)$ and $\mathsf{length}(O_m) = \mathsf{length}(O)$

$$\boxed{\begin{array}{lll} \mathsf{types}(I_1) = \cdots = \mathsf{types}(I_n) & = & \mathsf{types}(I) \quad \text{and} \\ \mathsf{types}(O_1) = \cdots = \mathsf{types}(O_n) & = & \mathsf{types}(O) \end{array}}$$

or, less restrictive,

$$\boxed{\begin{array}{lll} \mathsf{GCSub}\{\mathsf{types}(I_1), \ldots, \mathsf{types}(I_n)\} & = & \mathsf{types}(I) \quad \text{and} \\ \mathsf{LCSup}\{\mathsf{types}(O_1), \ldots, \mathsf{types}(O_n)\} & = & \mathsf{types}(O) \end{array}}$$

DSL for Network Specification (with types, no constraints)

$$\textsc{Let}\quad \frac{\Gamma \vdash \mathcal{M}_1 : (I_1, O_1) \ \cdots \ \Gamma \vdash \mathcal{M}_n : (I_n, O_n) \qquad \Gamma, X : (I, O) \vdash \mathcal{N} : (I', O')}{\Gamma \vdash \textbf{let } X \in \{\mathcal{M}_1, \ldots, \mathcal{M}_n\} \textbf{ in } \mathcal{N} \ : (I', O')}$$

for every $1 \leq m \leq n$

$\mathsf{parameters}(\mathcal{M}_m) \cap \mathsf{parameters}(\mathcal{N}) = \varnothing$

$\mathsf{length}(I_m) = \mathsf{length}(I)$ and $\mathsf{length}(O_m) = \mathsf{length}(O)$

$$\boxed{\begin{array}{l} \mathsf{types}(I_1) = \cdots = \mathsf{types}(I_n) \ = \ \mathsf{types}(I) \quad \text{and} \\ \mathsf{types}(O_1) = \cdots = \mathsf{types}(O_n) \ = \ \mathsf{types}(O) \end{array}}$$

or, less restrictive,

$$\boxed{\begin{array}{l} \mathsf{GCSub}\{\mathsf{types}(I_1), \ldots, \mathsf{types}(I_n)\} \ = \ \mathsf{types}(I) \quad \text{and} \\ \mathsf{LCSup}\{\mathsf{types}(O_1), \ldots, \mathsf{types}(O_n)\} \ = \ \mathsf{types}(O) \end{array}}$$

or, less restrictive still,

$$\boxed{\begin{array}{l} \mathsf{GCSub}\{\mathsf{types}(I_1), \ldots, \mathsf{types}(I_n)\} \ :> \ \mathsf{types}(I) \quad \text{and} \\ \mathsf{LCSup}\{\mathsf{types}(O_1), \ldots, \mathsf{types}(O_n)\} \ <: \ \mathsf{types}(O) \end{array}}$$

DSL for Network Specification (with types, no constraints)

$$\textsc{Let}\quad \frac{\Gamma \vdash \mathcal{M}_1 : (I_1, O_1) \ \cdots \ \Gamma \vdash \mathcal{M}_n : (I_n, O_n) \qquad \Gamma, X : (I, O) \vdash \mathcal{N} : (I', O')}{\Gamma \vdash \textbf{let } X \in \{\mathcal{M}_1, \ldots, \mathcal{M}_n\} \textbf{ in } \mathcal{N} \ : (I', O')}$$

for every $1 \leq m \leq n$

$\mathsf{parameters}(\mathcal{M}_m) \cap \mathsf{parameters}(\mathcal{N}) = \varnothing$

$\mathsf{length}(I_m) = \mathsf{length}(I)$ and $\mathsf{length}(O_m) = \mathsf{length}(O)$

$$\boxed{\begin{array}{lll} \mathsf{types}(I_1) = \cdots = \mathsf{types}(I_n) & = & \mathsf{types}(I) \quad \text{and} \\ \mathsf{types}(O_1) = \cdots = \mathsf{types}(O_n) & = & \mathsf{types}(O) \end{array}}$$

or, less restrictive,

$$\boxed{\begin{array}{lll} \mathsf{GCSub}\{\mathsf{types}(I_1), \ldots, \mathsf{types}(I_n)\} & = & \mathsf{types}(I) \quad \text{and} \\ \mathsf{LCSup}\{\mathsf{types}(O_1), \ldots, \mathsf{types}(O_n)\} & = & \mathsf{types}(O) \end{array}}$$

or, less restrictive still,

$$\boxed{\begin{array}{lll} \mathsf{GCSub}\{\mathsf{types}(I_1), \ldots, \mathsf{types}(I_n)\} & :> & \mathsf{types}(I) \quad \text{and} \\ \mathsf{LCSup}\{\mathsf{types}(O_1), \ldots, \mathsf{types}(O_n)\} & <: & \mathsf{types}(O) \end{array}}$$

Can we do better? Do these adjustments make rule Let compositional?

DSL for Network Specification (with types, no constraints)

Under an interpretation of "<:" that specifies that (*velocity intervals*)·(*density intervals*) can be safely:

- decreased at entering roads
- increased at exiting roads

$$\textsc{Subtyping} \qquad \frac{\Gamma \vdash \mathcal{M} : (I, O)}{\Gamma \vdash \mathcal{M} : (I', O')}$$

$$\mathsf{types}(I) :> \mathsf{types}(I')$$

$$\mathsf{types}(O) <: \mathsf{types}(O')$$

all three adjustments on preceding page are equivalent

DSL for Network Specification (with types, no constraints)

$$\textsc{Let}' \quad \frac{\Gamma \vdash \mathcal{M}_m : (I_m, O_m) \qquad \Gamma, X : (I_m, O_m) \vdash \mathcal{N} : (I'_m, O'_m) \qquad 1 \leq m \leq n}{\Gamma \vdash \textsf{\textbf{let}}\ X \in \{\mathcal{M}_1, \ldots, \mathcal{M}_n\}\ \textsf{\textbf{in}}\ \mathcal{N}\ : (I', O')}$$

$$\mathsf{parameters}(\mathcal{M}_n) \cap \mathsf{parameters}(\mathcal{N}) = \varnothing \quad \text{for all } 1 \leq m \leq n$$

$$\mathsf{length}(I_1) = \cdots = \mathsf{length}(I_n)$$

$$\mathsf{length}(O_1) = \cdots = \mathsf{length}(O_n)$$

$$I' = \mathsf{GCSub}\{I'_1, \ldots, I'_n\}$$

$$O' = \mathsf{LCSup}\{O'_1, \ldots, O'_n\}$$

DSL for Network Specification (with types, no constraints)

$$\textsc{Let}' \quad \frac{\Gamma \vdash \mathcal{M}_m : (I_m, O_m) \qquad \Gamma, X : (I_m, O_m) \vdash \mathcal{N} : (I'_m, O'_m) \qquad 1 \leq m \leq n}{\Gamma \vdash \textsf{\textbf{let}}\ X \in \{\mathcal{M}_1, \ldots, \mathcal{M}_n\}\ \textsf{\textbf{in}}\ \mathcal{N}\ : (I', O')}$$

$$\mathsf{parameters}(\mathcal{M}_n) \cap \mathsf{parameters}(\mathcal{N}) = \varnothing \quad \text{for all } 1 \leq m \leq n$$

$$\mathsf{length}(I_1) = \cdots = \mathsf{length}(I_n)$$

$$\mathsf{length}(O_1) = \cdots = \mathsf{length}(O_n)$$

$$I' = \mathsf{GCSub}\{I'_1, \ldots, I'_n\}$$

$$O' = \mathsf{LCSup}\{O'_1, \ldots, O'_n\}$$

FACT:

1. **Every well-typed network specification using rule $\textsc{Let}$ is also well-typed using rule $\textsc{Let}'$.**

2. **There are well-typed network specifications using rule $\textsc{Let}'$ that are not well-typed using rule $\textsc{Let}$.**

DSL for Network Specification (with types, no constraints)

$$\textsc{Let}' \quad \frac{\Gamma \vdash \mathcal{M}_m : (I_m, O_m) \qquad \Gamma, X : (I_m, O_m) \vdash \mathcal{N} : (I'_m, O'_m) \qquad 1 \leq m \leq n}{\Gamma \vdash \textsf{\textbf{let}}\ X \in \{\mathcal{M}_1, \ldots, \mathcal{M}_n\}\ \textsf{\textbf{in}}\ \mathcal{N}\ : (I', O')}$$

$$\mathsf{parameters}(\mathcal{M}_n) \cap \mathsf{parameters}(\mathcal{N}) = \varnothing \quad \text{for all } 1 \leq m \leq n$$

$$\mathsf{length}(I_1) = \cdots = \mathsf{length}(I_n)$$

$$\mathsf{length}(O_1) = \cdots = \mathsf{length}(O_n)$$

$$I' = \mathsf{GCSub}\{I'_1, \ldots, I'_n\}$$

$$O' = \mathsf{LCSup}\{O'_1, \ldots, O'_n\}$$

FACT:

1. **Every well-typed network specification using rule** $\textsc{Let}$ **is also well-typed using rule** $\textsc{Let}'$**.**

2. **There are well-typed network specifications using rule** $\textsc{Let}'$ **that are not well-typed using rule** $\textsc{Let}$**.**

Can we do better? Is rule $\textsc{Let}'$ compositional?

DSL for Network Specification (with types+constraints)

$$\textsc{Connect} \quad \frac{\Gamma \vdash \mathcal{M} : (I_1, O_1) \,|\, C_1 \qquad \Gamma \vdash \mathcal{N} : (I_2, O_2) \,|\, C_2}{\Gamma \vdash \textbf{conn}_{\theta,m,n}(\mathcal{M}, \mathcal{N}) : (I', O') \,|\, C}$$

$$\mathsf{parameters}(\mathcal{M}) \cap \mathsf{parameters}(\mathcal{N}) = \varnothing$$

$$m = \mathsf{length}(O_1) \qquad n = \mathsf{length}(I_2)$$

$$\theta : \subseteq_{1\text{-}1} \{1, \ldots, m\} \times \{1, \ldots, n\}$$

$$I = I_1 \cdot (I_2 / \mathsf{range}(\theta)) \qquad O = (O_1 / \mathsf{domain}(\theta)) \cdot O_2$$

$$[\mathsf{types}(O_1)]_p <: [\mathsf{types}(I_2)]_q \quad \text{for every } (p, q) \in \theta$$

$$\boxed{C \;=\; C_1 \cup C_2 \cup \{\text{“}[\mathsf{PP}(O_1)]_p \doteq [\mathsf{PP}(I_2)]_q\text{”} \mid (p, q) \in \theta\}}$$

If p-th outgoing link from $\mathcal{M}$ is connected
to q-th incoming link to $\mathcal{N}$ then
the parameter pairs of the two links are constrained to be equal.

Rule $\textsc{Loop}$ **adjusted in same way as rule** $\textsc{Connect}$**.**

Thank You!

Questions?

DSL for Network Specification (untyped)

For every θ_1, m_1, n_1 and θ_2, m_2, n_2
there are θ'_1, m'_1, n'_1 and θ'_2, m'_2, n'_2 such that

$$\mathbf{conn}_{\theta_1, m_1, n_1}(\mathcal{M}, \mathbf{conn}_{\theta_2, m_2, n_2}(\mathcal{N}, \mathcal{P})) =$$
$$\mathbf{conn}_{\theta'_1, m'_1, n'_1}(\mathbf{conn}_{\theta'_2, m'_2, n'_2}(\mathcal{M}, \mathcal{N}), \mathcal{P})$$

For every θ_1, m_1, n_1 and θ_2, m_2, n_2 and θ_3, m_3, n_3
there are θ'_1, m'_1, n'_1 and θ'_2, m'_2, n'_2 such that

$$\mathbf{conn}_{\theta_1, m_1, n_1}(\mathbf{loop}_{\theta_2, m_2, n_2}(\mathcal{M}), \mathbf{loop}_{\theta_3, m_3, n_3}(\mathcal{N})) =$$
$$\mathbf{loop}_{\theta'_1, m'_1, n'_1}(\mathbf{conn}_{\theta'_2, m'_2, n'_2}(\mathcal{M}, \mathcal{N}))$$

DSL for Network Specification (untyped)

Useful special cases

$$\textsc{Connect}^{n} \qquad \frac{\Gamma \vdash \mathcal{M} : (I_1, O_1) \qquad \Gamma \vdash \mathcal{N} : (I_2, O_2)}{\Gamma \vdash \mathcal{M} \boxed{n} \mathcal{N} : (I_1, O_2)}$$

$\textsf{parameters}(\mathcal{M}) \cap \textsf{parameters}(\mathcal{N}) = \varnothing$
$\mathcal{M} \boxed{n} \mathcal{N} = \textbf{conn}_{\theta,n,n}(\mathcal{M}, \mathcal{N})$
$\textsf{length}(O_1) = \textsf{length}(I_2) = n \geq 1$
$\theta = \{(1,1), (2,2), \ldots, (n,n)\}$

$$\textsc{Connect}^{\boxtimes} \qquad \frac{\Gamma \vdash \mathcal{M} : (I_1, O_1) \qquad \Gamma \vdash \mathcal{N} : (I_2, O_2)}{\Gamma \vdash \mathcal{M} \boxtimes \mathcal{N} : (I_1, O_2)}$$

$\textsf{parameters}(\mathcal{M}) \cap \textsf{parameters}(\mathcal{N}) = \varnothing$
$\mathcal{M} \boxtimes \mathcal{N} = \textbf{conn}_{\theta,2,2}(\mathcal{M}, \mathcal{N})$
$\textsf{length}(O_1) = \textsf{length}(I_2) = 2$
$\theta = \{(1,2), (2,1)\}$

$$\textsc{Connect}^{\parallel} \qquad \frac{\Gamma \vdash \mathcal{M} : (I_1, O_1) \qquad \Gamma \vdash \mathcal{N} : (I_2, O_2)}{\Gamma \vdash \mathcal{M} \parallel \mathcal{N} : (I, O)}$$

$\textsf{parameters}(\mathcal{M}) \cap \textsf{parameters}(\mathcal{N}) = \varnothing$
$\mathcal{M} \parallel \mathcal{N} = \textbf{conn}_{\theta,m,n}(\mathcal{M}, \mathcal{N})$
$\textsf{length}(O_1) = m \qquad \textsf{length}(I_2) = n \qquad \theta = \varnothing$
$I = I_1 \cdot I_2 \qquad O = O_1 \cdot O_2$

Useful special cases for rule $\textsc{Loop}$ **as for rule** $\textsc{Connect}$**.**

More Formal Methods ...

for the development of a rigorous discipline of *specification*, *analysis*, *programming* and *maintenance* of network systems

2. An Application of Model Checking:
Safe Composition of Arbitrary Network Protocols
(mostly with **Adam Bradley** and **Azer Bestavros**)
iBench Initiative – http://www.cs.bu.edu/groups/ibench/

3. Resource Allocation in Sensor Networks Using
a Strongly-Typed Domain-Specific Language
(mostly with **Michael Ocean** and **Azer Bestavros**)
snBench – http://csr.bu.edu/snbench/

4. The Stable-Paths Problem and the Promise of
an Automatic Lightweight Proof-Assistant
(mostly with **Kevin Donnelly** and **Andrei Lapets**)

Lightweight Formal Methods for the Development of High-Assurance Network Systems

Assaf Kfoury

with contributions from

Azer Bestavros, Adam Bradley, Andrei Lapets, and Michael Ocean

iBench Initiative
http://www.cs.bu.edu/groups/ibench/
snBench
http://csr.bu.edu/snbench/

Computer Science