Relating State-Based and Process-Based Concurrency through Linear Logic

Iliano Cervesato

Carnegie Mellon University - Qatar

iliano@cmu.edu

École Polytechnique
17 September 2009

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Specifying Concurrent Systems

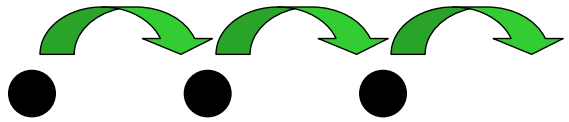
- Two main approaches
  - + Transition-based
    - ➢ Petri nets, multiset rewriting, …
  - + Process-based
    - ➢ Process algebras, …
- No language supports both
  - + Different linguistic features
  - + Different analysis methods
- Ad hoc translations
- Concurrency inherent to many problems
  - + Cryptographic protocols
  - + ...

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

State-Based vs. Process-Based

- State-based languages
  - Multiset Rewriting
  - NRL Prot. Analyzer, CAPSL/CIL, Paulson's approach, …

  + State transition semantics



- Process-based languages
  - Process Algebra
  - Strand spaces, spi-calculus, …

  + Independent communicating threads

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

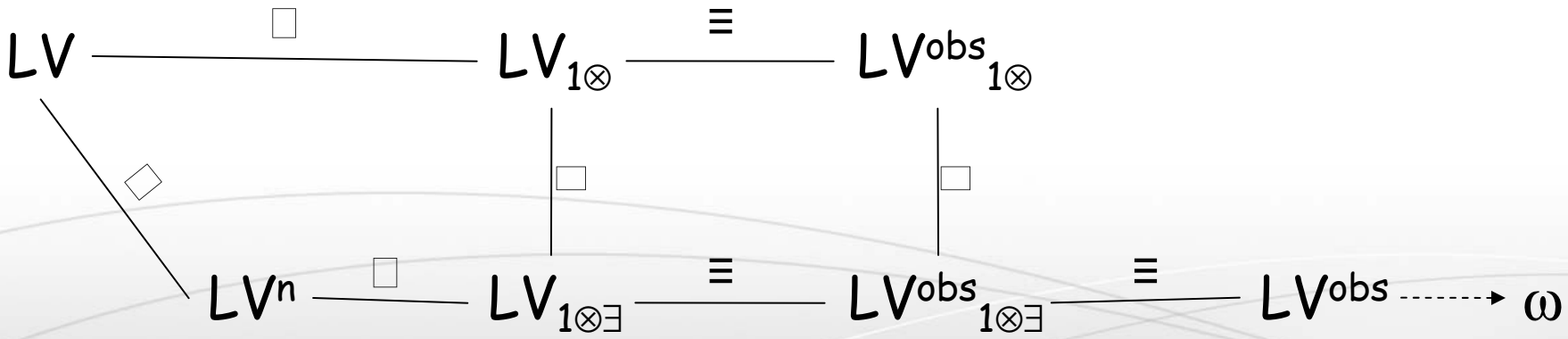
Summary of Results

- System ω
  - + Rationalization of multiset rewriting
    - ➢ Traditional multiset rewriting are sublanguages
    - ➢ Simpler, but much more expressive
  - + Significant bridge to process algebras
    - ➢ Popular algebras are sublanguages
  - + Both in the same seamless formalism
  - + Proof-theoretic foundations
- MSR 3
  - + Specialization of ω to security protocols

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Methodology

- Rewriting re-interpretation of linear logic
  - + Open derivations
  - + Left-rule semantics
- Successive refinements of LV sequent system



Further Developments

- Verifying specifications
  - + Transferring methods
    - ➢ Equivalence, bisimulation in ω
  - + Model checking
- Additional application domains
  - + Massively distributed systems
    - ➢ Claytronics
  - + Molecular biology
    - ➢ Modeling cellular pathways
  - + Micro-economic simulation
    - ➢ Predicting effect of policies

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Logical Foundations

- Linear logic in LV
- Tensorial observations – $LV^{obs}_{1\otimes}$
- Tensorial-existential observations – $LV^{obs}_{1\otimes\exists}$
- Cut-elimination
- Rewriting interpretation
- The system ω

Linear Logic

- Formulas

  A, B ::= a | 1 | A ⊗ B | A —o B | ! A
  | T | A & B | ∀x. A | ∃x. A

- LV sequents

  - Constructor: ","
  - Empty: "•"

$$\Gamma ; \Delta \xrightarrow[\Sigma]{} C$$

Unrestricted context

Linear context

Signature

Goal formula

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Some LV Rules

Structural rules

$$\frac{}{\Gamma;\ A \dashrightarrow_\Sigma A} \qquad \frac{\Gamma,\ A;\ \Delta\ ,\ A \dashrightarrow_\Sigma C}{\Gamma,\ A;\ \Delta \dashrightarrow_\Sigma C}$$

Left rules

$$\frac{\Gamma;\ \Delta,\ A,\ B \dashrightarrow_\Sigma C}{\Gamma;\ \Delta,\ A \otimes B \dashrightarrow_\Sigma C}$$

$$\frac{\Gamma;\ \Delta' \dashrightarrow_\Sigma A \quad \Gamma;\ \Delta,\ B \dashrightarrow_\Sigma C}{\Gamma;\ \Delta,\ \Delta'\ ,\ A \multimap B \dashrightarrow_\Sigma C}$$

$$\frac{\Sigma \vdash t \quad \Gamma;\ \Delta,\ [t/x]A \dashrightarrow_\Sigma C}{\Gamma;\ \Delta,\ \forall x.A \dashrightarrow_\Sigma C}$$

$$\frac{\Gamma;\ \Delta,\ A \dashrightarrow_{\Sigma,x} C}{\Gamma;\ \Delta,\ \exists x.A \dashrightarrow_\Sigma C}$$

$$\frac{\Gamma,\ A;\ \Delta \dashrightarrow_\Sigma C}{\Gamma;\ \Delta\ ,\ !A \dashrightarrow_\Sigma C}$$

...

Cut rules

$$\frac{\Gamma;\ \Delta' \dashrightarrow_\Sigma A \quad \Gamma;\ \Delta,\ A \dashrightarrow_\Sigma C}{\Gamma;\ \Delta,\Delta' \dashrightarrow_\Sigma C}$$

$$\frac{\Gamma;\ \bullet \dashrightarrow_\Sigma A \quad \Gamma,\ A;\ \Delta \dashrightarrow_\Sigma C}{\Gamma;\ \Delta \dashrightarrow_\Sigma C}$$

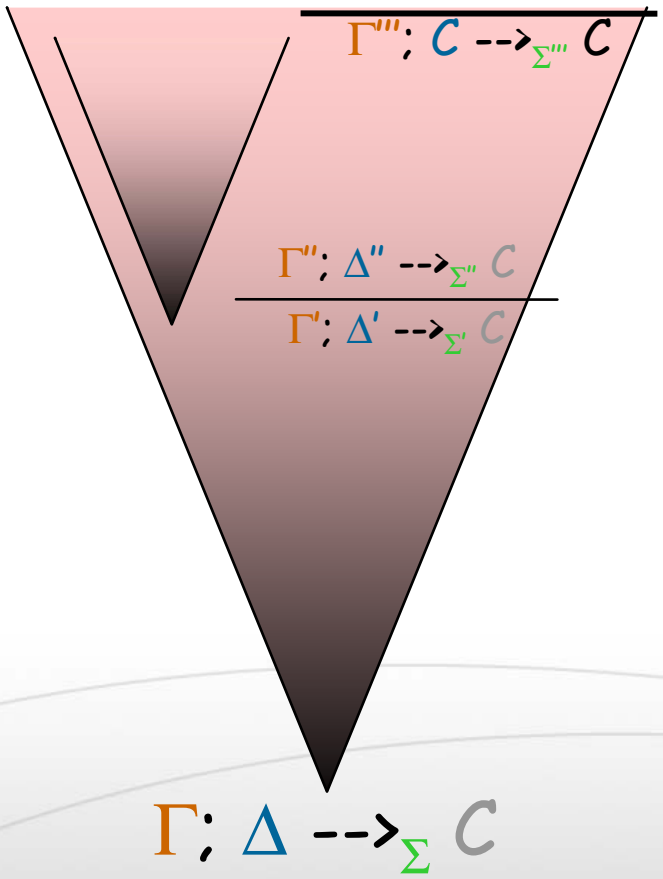
Right rules

$$\frac{}{\Gamma;\ \Delta \dashrightarrow_\Sigma 1} \qquad \frac{\Gamma;\ \Delta_1 \dashrightarrow_\Sigma C_1 \quad \Gamma;\ \Delta_2 \dashrightarrow_\Sigma C_2}{\Gamma;\ \Delta_1,\ \Delta_2 \dashrightarrow_\Sigma C_1 \otimes C_2}$$

$$\frac{\Sigma \vdash t \quad \Gamma;\ \Delta \dashrightarrow_\Sigma [t/x]C}{\Gamma;\ \Delta \dashrightarrow_\Sigma \exists x.C}$$

...

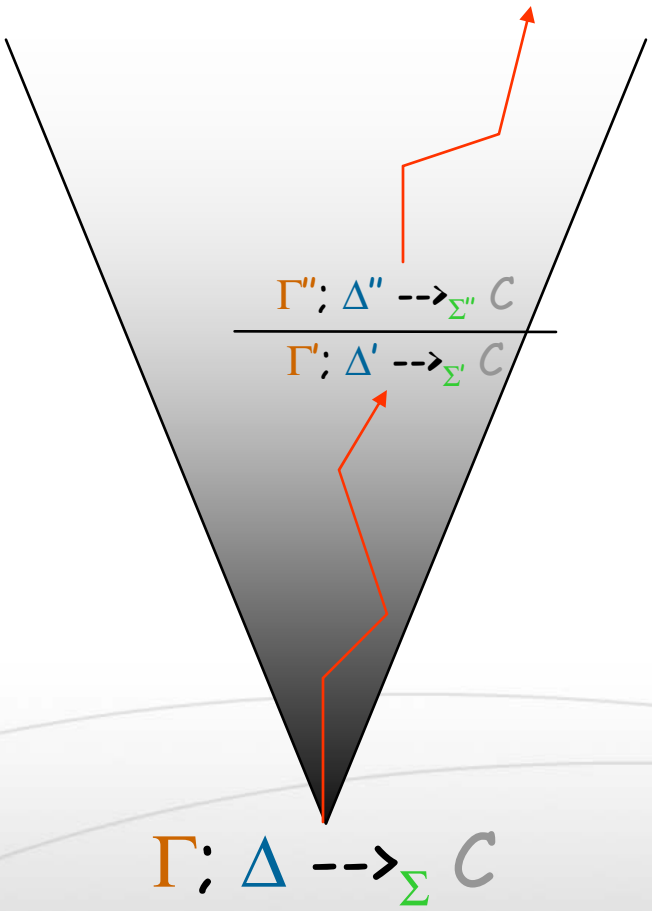
Logical Derivations



- Proof of C from Δ and Γ
  - + Emphasis on C
    - ➢ C is input
- Finite
  - + Closed
- Rules shown
  - + Major premise
    - ➢ Preserves C
  - + Minor premise
    - ➢ Starts subderivation

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

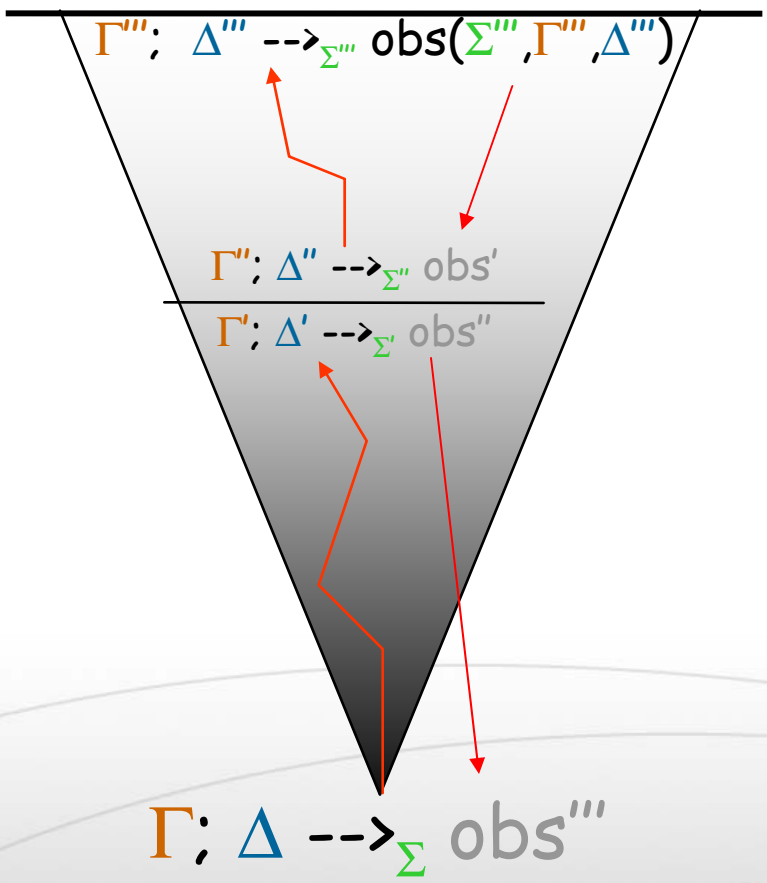
A Rewriting Re-Interpretation



- Transition
  - ➢ From conclusion
  - ➢ To major premise
  - \+ Emphasis on Γ, Δ and Σ
  - \+ C is output, at best
    - ➢ Does not change
- Possibly infinite
  - \+ Open
- Minor premise
  - \+ Auxiliary rewrite chain
    - ➢ Finite
  - \+ Topped with axiom

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Observations



- Close derivation to "observe" Γ, Δ and Σ
  - + At any point
- Use C to propagate observation to top level
- How to engineer this?
  - + Restrict right rules (drammatically)
  - + Get rid of binary rules

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Tensorial observations – $LV^{obs}_{1\otimes}$

- True in LV:
  - + Monoidal properties of *1* and $\otimes$
  - + Γ; Δ -->$_\Sigma$ $\otimes\Delta$
  - + Γ; Δ -->$_\Sigma$ C *iff* Γ; $\otimes\Delta$ -->$_\Sigma$ C
- $LV_{1\otimes}$: remove all right rules except for *1* and $\otimes$
  - + Expressiveness limited to collecting context
- $LV^{obs}_{1\otimes}$: replace right rules and id with

$$\frac{}{\Gamma;\ \Delta \dashrightarrow_\Sigma \otimes\Delta}$$

+ Γ; Δ -->$_\Sigma$ C in $LV_{1\otimes}$ *iff* Γ; Δ -->$_\Sigma$ C in $LV^{obs}_{1\otimes}$

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Nominal Quantification

$$\frac{\Gamma;\ \Delta \dashrightarrow_{\Sigma,x} C}{\Gamma;\ \Delta \dashrightarrow_{\Sigma,x} \exists x.C}$$

- Binds all occurrences of x in C
  - + Reification of a sequent-level binder
- All interpretations of concurrent languages rely on it
  - + Often unknowningly

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Nominal Observations – $LV^{obs}_{1\otimes\exists}$

- True in LV:
  - + Mobility laws
  - + $\Gamma; \Delta \dashrightarrow_{\Sigma,\Sigma'} \exists\Sigma'.\Delta$
  - + $\Gamma; \Delta \dashrightarrow_{\Sigma,\Sigma'} C$ *iff* $\Gamma; \exists\Sigma'.\Delta \dashrightarrow_{\Sigma} C$ if $\Sigma' \cap FV(\Gamma,C) = \varnothing$
- $LV_{1\otimes}$: remove all right rules except 1, $\otimes$ and $\exists$
  - + Expressiveness limited to collecting context
- $LV^{obs}_{1\otimes\exists}$: replace right rules and id with

$$\frac{}{\Gamma; \Delta \dashrightarrow_{\Sigma,\Sigma'} \exists\Sigma'.\Delta}$$

  - + If $\Gamma; \Delta \dashrightarrow_{\Sigma} C$ *in* $LV_{1\otimes\exists}$, then $C \equiv \exists\Sigma'.\Delta'$ and $\Gamma; \Delta \dashrightarrow_{\Sigma} \exists\Sigma'.\Delta'$ in $LV^{obs}_{1\otimes\exists}$
  - + If $\Gamma; \Delta \dashrightarrow_{\Sigma} C$ in $LV^{obs}_{1\otimes\exists}$, *then* $\Gamma; \Delta \dashrightarrow_{\Sigma} C$ *in* $LV_{1\otimes\exists}$

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Structural Equivalences

Monoidal laws

+ $A \otimes B = B \otimes A$
+ $A \otimes 1 = A$
+ $(A \otimes B) \otimes C = A \otimes (B \otimes C)$

Mobility laws

+ $\exists x.\, \exists y.\, A = \exists y.\, \exists x.\, A$
+ $\exists x.\, 1 = 1$
+ $\exists x.\, (A \otimes B) = A \otimes \exists x.\, B$
  if $x \notin FV(A)$

- Logical bi-equivalences
  - Require limited right rules
- Express structure of context / binders

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Dealing with Binary Rules

- Implication
  - + Inline side-derivation
  - + LV^{obs}: replace right rule for $\multimap$ with

$$\frac{\Gamma;\ \Delta,\ B \dashrightarrow_{\Sigma,\Sigma'} C}{\Gamma;\ \Delta,\ \Delta',\ \exists\Sigma'.\Delta' \multimap B \dashrightarrow_{\Sigma,\Sigma'} C}$$

  - + $\Gamma;\ \Delta \dashrightarrow_{\Sigma} C$ in $LV^{obs}{}_{1\otimes\exists}$ *iff* $\Gamma;\ \Delta \dashrightarrow_{\Sigma} C$ in LV^{obs}
- Cut
  - + Cut rules are admissible
    - ➢Simplified adaptation of usual proof

Rewriting Interpretation of LV^{obs}

- All rules are unary
  - + Except observation rule
- States

$$\Sigma \;;\; \Gamma \;;\; \Delta$$

  - + Σ is a list
  - + Γ and Δ are commutative monoids
  - + No C
    - ➢ Does not change
- Transitions

$$\Sigma; \Gamma; \Delta \rightarrow \Sigma'; \Gamma'; \Delta'$$

- ➢ Constructor: ","
- ➢ Empty: "•"

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

LV^{obs} Rules as Rewrite Rules

—o	(Σ, Σ') ; Γ ; $(\Delta, \Delta', (\Sigma'.\Delta' \multimap B))$	$\rightarrow (\Sigma, \Sigma')$; Γ ; (Δ, B)
T		(no rules)
&	Σ ; Γ ; $(\Delta, A_1 \,\&\, A_2)$	$\rightarrow \Sigma$; Γ ; (Δ, A_i)
$\forall$	Σ ; Γ ; $(\Delta, \forall x.\, A)$	$\rightarrow \Sigma$; Γ ; $(\Delta, [t/x]A)$ if $\Sigma \vdash t$
!	Σ ; Γ ; $(\Delta, !A)$	$\rightarrow \Sigma$; (Γ, A) ; Δ
	Σ ; (Γ, A) ; Δ	$\rightarrow \Sigma$; (Γ, A) ; (Δ, A)
1	Σ ; Γ ; $(\Delta, 1)$	$\rightarrow \Sigma$; Γ ; Δ
$\otimes$	Σ ; Γ ; $(\Delta, A \otimes B)$	$\rightarrow \Sigma$; Γ ; (Δ, A, B)
$\exists$	Σ ; Γ ; $(\Delta, \exists x.\, A)$	$\rightarrow (\Sigma, x)$; Γ ; (Δ, A)

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Formal Correspondence

- Wrt LV^{obs}
  - If Σ ; Γ ; Δ $\rightarrow^*$ (Σ,Σ'); (Γ,Γ') ; Δ',
    then Γ; Δ $\dashrightarrow_{\Sigma}$ $\exists\Sigma'.\Delta'$ *in* LV^{obs}
  - If Γ; Δ $\dashrightarrow_{\Sigma}$ C *in* LV^{obs},
    then $C \equiv \exists\Sigma'.\Delta'$ and Σ ; Γ ; Δ $\rightarrow^*$ (Σ,Σ'); (Γ,Γ') ; Δ'
- Wrt LV
  - Sound
  - Not complete
    - No! We have only crippled right rules

      • ; • ; a —o b, b —o c $\rightarrow^*$ • ; a —o c

System ω

- Monoidal equivalences allow identifying
  - + ⊗ with linear context constructor ","
  - + **1** with empty linear context "•"
- Correspondence with logic
  - + If Σ ; Γ ; Δ $\rightarrow^*$ (Σ,Σ'); (Γ,Γ') ; Δ',
    then Γ; Δ $\dashrightarrow_{\Sigma}$ $\exists\Sigma'.\Delta'$ *in* LV^{obs}
  - + If Γ; Δ $\dashrightarrow_{\Sigma}$ C *in* LV^{obs},
    then $C \equiv \exists\Sigma'.\Delta'$ and Σ ; Γ ; Δ $\rightarrow^*$ (Σ,Σ'); (Γ,Γ') ; Δ'

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

ω–multisets

A, B	::=	a	atomic object
	|	•	empty
	|	A , B	formation
	|	A —o B	rewrite
	|	T	no-op
	|	A & B	choice
	|	∀x. A	instantiation
	|	∃x. A	generation
	|	! A	replication

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Discussion

- Other connectives?
  - + $\oplus$, 0, $\wp$, $\bot$
    - ➢ Odd rewrite properties
  - + ?, $(\_)^{\bot}$
    - ➢ Not yet explored
- Other presentations?
- Other logics?
  - + Very close to CLF
- Other forms of proof-as-computation?
  - + Dual of logic programming
  - + Similar to ACL [Kobayashi & Yonezawa, 93]
- Can logic benefit?

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Multiset Rewriting

- Multiset: set with repetitions allowed

  a ::= • | a, a

  + Commutative monoid

- Multiset rewriting (a.k.a. Petri nets)
  + Rewriting within the monoid
  + Fundamental model of distributed computing
    - Alternative: Process Algebras
  + Basis for security protocol spec. languages
    - MSR family
    - … several others
  + Many extensions, more or less ad hoc

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

First-Order Multiset Rewriting

- Multiset elements are F0 atomic formulas
- Rules have the form

$$\forall x_1 \ldots x_n.\ \underline{a}(\mathbf{x}) \rightarrow \exists y_1 \ldots y_k.\ \underline{b}(\mathbf{x},\mathbf{y})$$

- Semantics

$$\Sigma\ ;\ \underline{a}(t),\ \underline{s} \rightarrow_{R,\ (\underline{a}(x) \rightarrow \exists y.\ \underline{b}(x,y))} \Sigma,y\ ;\ \underline{b}(t,y),\ \underline{s} \quad \text{if } \Sigma \vdash t$$

- Several encodings into linear logic
  - + [Martí-Oliet, Meseguer, 91]

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

ω-Multisets vs. Multiset Rewriting

- MSR 1 is an instance of ω-multisets
  - Uses only ⊗, 1, ∀, ∃, and —o
  - —o never nested, always persistent

+ $\Sigma ; \underline{s} \rightarrow_R \Sigma' ; \underline{s}'$
  iff $\Sigma ; \text{“R”} ; \text{“}\underline{s}\text{”} \rightarrow^* \Sigma' ; \text{“}\underline{s}'\text{”}$

- Interpretation of MSR **as** linear logic
  + Logical explanation of multiset rewriting
    - MSR is logic
  + Guideline to design rewrite systems

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

The Asynchronous π-Calculus

Another fundamental model of distributed computing

- Language

  P ::= 0 | P||Q | ν x. P | !P | x(y).P | $\underline{x}$<y>

- Semantics

  + Structural equivalence

    - Comm. monoidal congruence of || and 0
    - Binder mobility congruence of ν
      - ν x. ν y. P ≡ ν y. ν x. P
      - 0 ≡ ν x. 0
      - P || ν x. Q ≡ ν x. (P || Q) if x ∉ FN(P)

  + Reaction law

    - $\underline{x}$<y> || x(z). P || Q => [y/z]P || Q
    - !P => !P || P

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

π-calculus in ω-Multisets

- $0 \Leftrightarrow 1$
- $\| \Leftrightarrow \otimes$
- $\nu \Leftrightarrow \exists$
- $! \Leftrightarrow !$
- $x(y).\ P \Leftrightarrow \forall y.\ ch(x,y) \multimap$ "P"
- $\underline{x}\langle y \rangle \Leftrightarrow ch(x,y)$

- Reaction law
  - $\Sigma; \Gamma; ch(x,y), \forall z.\ ch(x,z) \multimap P, \Delta \rightarrow^2 \Sigma; \Gamma; [y/z]P, \Delta$
- Structural equivalence
  - Monoidal congr. of $\|$ and $0 \Leftrightarrow$ monoidal laws of $\otimes$ and 1
  - Mobility congr. of $\nu \Leftrightarrow$ mobility laws of $\exists$
  - $!P \equiv !P \| P$
    - Only $\Rightarrow$ in ω-multisets

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Properties

- If P =>* Q,
  then Σ_P; •; "P" →* (Σ_P, Σ); Γ; Δ
  where "Q" = $\exists\Sigma.\ !\Gamma \otimes \Delta$
- If Σ_P; •; "P" →* (Σ_P, Σ); Γ; Δ,
  then there exists Q such that "Q" = $\exists\Sigma.\ !\Gamma \otimes \Delta$
  and P =>* Q

ω-Multisets vs. Process Algebra

- Simple encoding of asynchronous π-calculus into ω-multisets
  - + Doesn't show that π-calculus is logic
  - + Uses only a fraction of ω-multiset syntax
  - + Inverse encoding?
    - ➢ As hard as going from multiset rewriting to π-calculus
- Other languages
  - + Join calculus
  - + Strand spaces
  - + To do: Synchronous π-calculus

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

MSR 3

- Instance of ω-multisets for cryptographic protocol specification
  - + Security-relevant signature
  - + Typing infrastructure
  - + Modules, equations, …
- 3rd generation
  - + MSR 1: First-order multiset rewriting with ∃
    - ➢ Undecidability of protocol analysis
  - + MSR 2: MSR 1 + typing
    - ➢ Actual specification language
    - ➢ More theoretical results

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Example

Needham-Schroeder public-key protocol

- $A \rightarrow B$: $\{n_A, A\}_{kB}$
- $B \rightarrow A$: $\{n_A, n_B\}_{kA}$
- $A \rightarrow B$: $\{n_B\}_{kB}$

- Can be expressed in several ways
  - + State-based
    - ➢ Explicit local state
    - ➢ As in MSR 2
  - + Process-based: embedded $\rightarrow$
    - ➢ Continuation-passing style
    - ➢ As in process algebra
  - + (Intermediate approaches)

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

$A \rightarrow B: \{n_A, A\}_{kB}$
$B \rightarrow A: \{n_A, n_B\}_{kA}$
$A \rightarrow B: \{n_B\}_{kB}$

State-Based

MSR 2 spec.

$\forall A$: princ.

{ $\exists L$: princ $\times$ $\sum B$:princ.pubK B $\times$ nonce $\rightarrow$ mset.

$\forall B$: princ. $\forall k_B$: pubK B.

•

$\rightarrow \exists n_A$: nonce.

net $(\{n_A, A\}_{kB})$, L (A, B, k_B, n_A)

$\forall B$: princ. $\forall k_B$: pubK B.
$\forall k_A$: pubK A. $\forall k_A'$: prvK k_A.
$\forall n_A$: nonce. $\forall n_B$: nonce.

net $(\{n_A, n_B\}_{kA})$, L (A, B, k_B, n_A)

$\rightarrow$ net $(\{n_B\}_{kB})$

}

Interpretation of L

- Rule invocation
  - Implementation detail
  - Control flow
- Local state of role
  - Explicit view
  - Important for DOS

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

$A \rightarrow B: \{n_A, A\}_{kB}$
$B \rightarrow A: \{n_A, n_B\}_{kA}$
$A \rightarrow B: \{n_B\}_{kB}$

Process-Based

$\forall A$:princ.

$\forall B$: princ. $\forall k_B$: pubK B.

- $\rightarrow$ $\exists n_A$: nonce.

  net $(\{n_A, A\}_{kB})$,

  ($\forall k_A$: pubK A. $\forall k_A'$: prvK k_A. $\forall n_B$: nonce.

  net $(\{n_A, n_B\}_{kA}) \rightarrow$ net $(\{n_B\}_{kB})$)

- Succinct
- Continuation-passing style
  - Rule asserts what to do next
  - Lexical control flow
- State is implicit
  - Abstract

NSPK in Process Algebra

$A \rightarrow B: \{n_A, A\}_{kB}$
$B \rightarrow A: \{n_A, n_B\}_{kA}$
$A \rightarrow B: \{n_B\}_{kB}$

$\forall$A:princ.

$\forall$B: princ. $\forall k_B$: pubK B.

$\forall k_A$: pubK A. $\forall k_A'$: prvK k_A. $\forall n_B$: nonce.

νn_A: nonce.

net $(\{n_A, A\}_{kB})$.

net $<\{n_A, n_B\}_{kA}>$.

net $(\{n_B\}_{kB})$. 0

Same structure !

- Not a coincidence
- MSR 3 very close to Process Algebra
  - ω-multiset encodings of π-calculus and Join Calculus

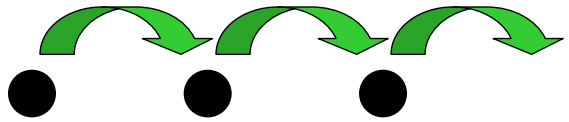
- MSR 3 is promising middle-ground for relating
  - State-based
  - Process-based

representations of a problem

State-Based vs. Process-Based

- State-based languages
  - Multiset Rewriting
  - NRL Prot. Analyzer, CAPSL/CIL, Paulson's approach, …

  + State transition semantics



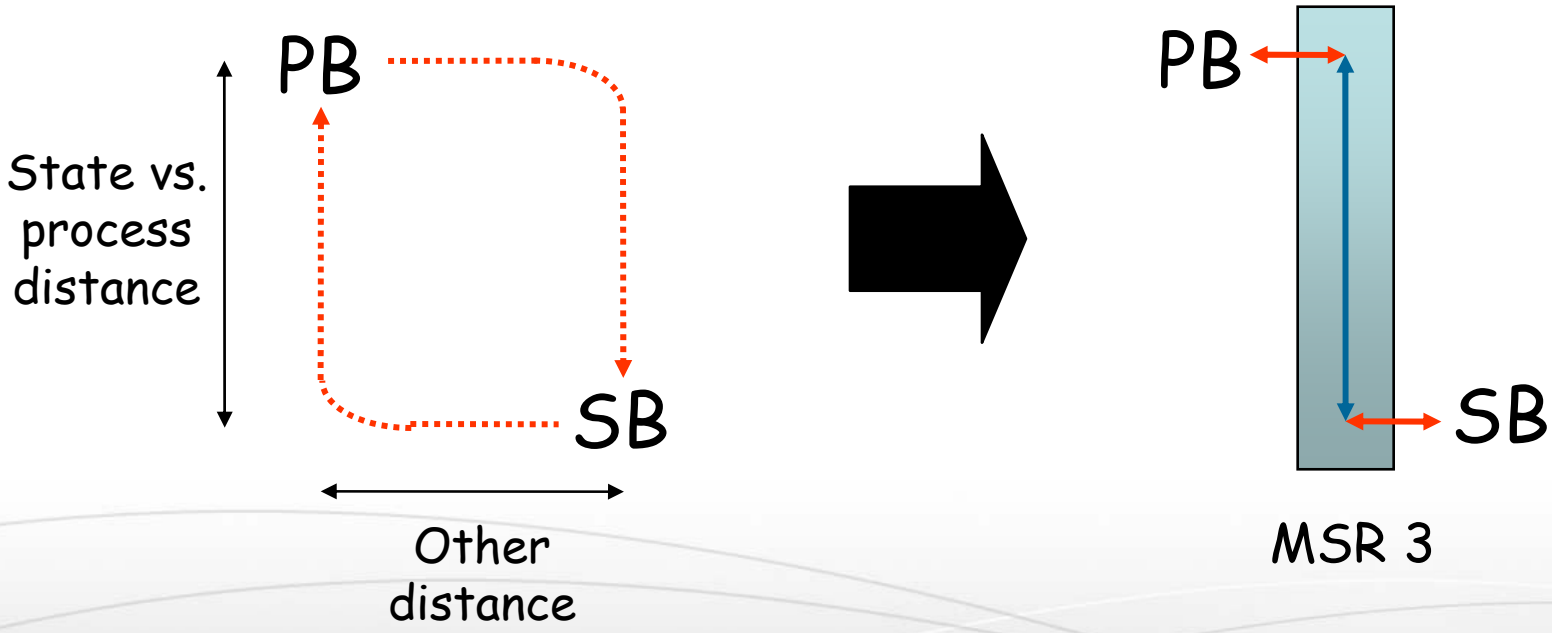
- Process-based languages
  - Process Algebra
  - Strand spaces, spi-calculus, …

  + Independent communicating threads

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

MSR 3 Bridges the Gap

- Difficult to go from one to the other
  - + Different paradigms



State ↔ Process translation done once and for all in MSR 3

Conclusions

- ω-multisets
  - + Logical foundation of multiset rewriting
  - + Relationship with process algebras
  - + Unified logical view
    - ➢ Better understanding of where we are
    - ➢ Hint about where to go next
- MSR 3.0
  - + Language for security protocol specification
  - + Succinct representations
    - ➢ Simpler specifications
    - ➢ Economy of reasoning
  - + Bridge between
    - ➢ State-based representation
    - ➢ Process-based representation

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Interpreting Unary Rules

$$\frac{\Gamma;\ \Delta,\ A,\ B \dashrightarrow_{\Sigma} C}{\Gamma;\ \Delta,\ A \otimes B \dashrightarrow_{\Sigma} C}$$

$$\Sigma;\ \Gamma;\ (\Delta,\ A \otimes B\ ) \rightarrow \Sigma;\ \Gamma;\ (\Delta,\ A,\ B)$$

$$\frac{\Sigma \vdash t \quad \Gamma;\ \Delta,\ [t/x]A \dashrightarrow_{\Sigma} C}{\Gamma;\ \Delta,\ \forall x.A \dashrightarrow_{\Sigma} C}$$

$$\Sigma;\ \Gamma;\ (\Delta,\ \forall x.\ A) \rightarrow \Sigma;\ \Gamma;\ (\Delta,\ [t/x]A) \quad \text{if } \Sigma \vdash t$$

$$\frac{\Gamma;\ \Delta,\ A \dashrightarrow_{\Sigma,x} C}{\Gamma;\ \Delta,\ \exists x.A \dashrightarrow_{\Sigma} C}$$

$$\Sigma;\ \Gamma;\ (\Delta,\ \exists x.\ A) \rightarrow (\Sigma,\ x);\ \Gamma;\ (\Delta, A)$$

$$\frac{\Gamma,\ A;\ \Delta \dashrightarrow_{\Sigma} C}{\Gamma;\ \Delta\ ,\ !A \dashrightarrow_{\Sigma} C}$$

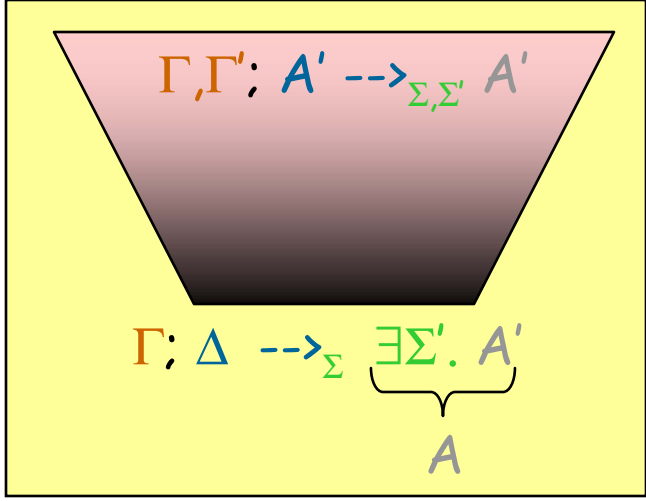
$$\Sigma;\ \Gamma;\ (\Delta,\ !A) \rightarrow \Sigma;\ (\Gamma,\ A);\ \Delta$$

...

...

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Observations



- Observation states

  $\Sigma \;;\; \Delta$

  + In Δ, we identify
    - , with $\otimes$
    - • with 1

    Categorical semantics

  $\Delta = \otimes\Delta$

  + Identified with $\exists x_1. \ldots \exists x_n. \Delta$
    - For $\Sigma = x_1, \ldots, x_n$

    De Bruijn's telescopes

  $\Sigma; \Delta = \exists\Sigma. \otimes\Delta$

- Observation transitions

  $\Sigma; \Gamma; \Delta \rightarrow^* \Sigma'; \Delta'$

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar

Type Theoretic Side

- Very close to CLF

  Concurrent Logical Framework

  + Linear type theory with
    - Dependent function types: Π (LF)
    - Asynchronous connectives: $\multimap$, &, T (LLF)
    - Synchronous connectives: $\otimes$, 1, !, $\exists$
    - Monadic sandboxing
    - Concurrency equations
  + Faithful encoding of true concurrency
    - Petri nets, MSR 2 specs, π-calculus, concurrent ML

- Details of relation still unclear

جامعة كارنيجي ميلون في قطر
Carnegie Mellon Qatar