MSR 3:

One Year Later

MSR 3.0:

The Logical Meeting Point of Multiset Rewriting and Process Algebra

Iliano Cervesato iliano@itd.nrl.navy.mil

ITT Industries, inc @ NRL Washington, DC

http://www.cs.stanford.edu/~iliano

CS Department, UMBC February 27-28, 2003

Iliano Cervesato iliano@itd.nrl.navy.mil

ITT Industries, inc @ NRL Washington, DC

http://theory.stanford.edu/~iliano

NSPK in MSR 3

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

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

NSPK in MSR 3

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

Not an MSR 2 spec.

$\forall A$: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

Looks Familiar?

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

Process calculus

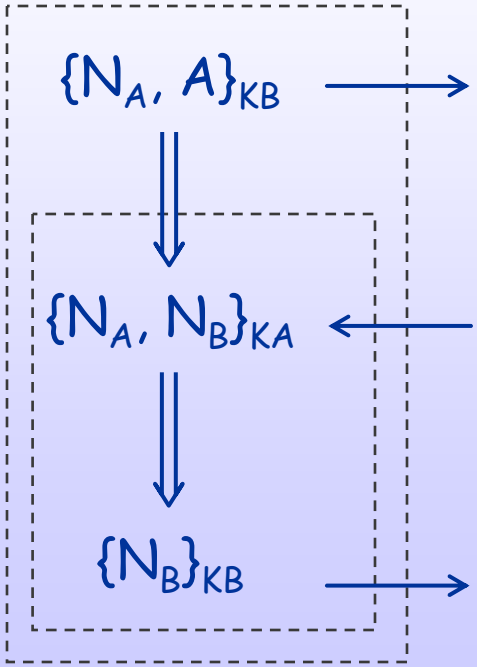
$\forall A$:princ.
$\forall B$: princ. $\forall k_B$: pubK B.
$\forall k_A$: pubK A. $\forall k_A'$: prvK k_A. $\forall n_B$: nonce.

νn_A: nonce.
net $(\{n_A, A\}_{kB})$.
net $\langle\{n_A, n_B\}_{kA}\rangle$.
net $(\{n_B\}_{kB})$. 0

Parametric strand

Alice (A,B,N_A,N_B) :

N_A Fresh, π_A (A,B)



What is MSR 3?

A new language for security protocols

- Supports
  - State transition specs
    - Conservative over MSR 2
  - Process algebraic specs

  } Neutral paradigm

- Rewriting re-interpretation of logic
  - Rich composable set of connectives
- Universal connector

More than the Sum of its Parts

Process- and transition-based specs.
in the **same** language

- Choose the paradigm
  - User's preference
  - Highlight characteristics of interest
  - Support various verification techniques (FW)
- Mix and match styles
  - Within a spec.
  - Within a protocol
  - Within a role

What is in MSR 3 ?

- Security-relevant signature
  - Network
  - Encryption, …
- Typing infrastructure
  - Dependent types
  - Subsorting
- Data Access Specification (DAS)
- Module system
- Equations

From MSR 1

From MSR 2

From MSR 2 implementation

ω-Multisets

Specification language for concurrent systems

- Crossroad of
  - State transition languages
    - Petri nets, multiset rewriting, …
  - Process calculi
    - CCS, π-calculus, …
  - (Linear) logic
- Benefits
  - Analysis methods from logic and type theory
  - Common ground for comparing
    - Multiset rewriting
    - Process algebra
  - Allows multiple styles of specification
    - Unified approach

Syntax

A	::=	a		atomic object
	|	1	$[\bullet]$	empty
	|	$A \otimes B$	$[A, B]$	formation
	|	$A \multimap B$	$[A \to B]$	rewrite
	|	$\top$		no-op
	|	$A \,\&\, B$	$[A \,\|\|\, B]$	choice
	|	$\forall x.\ A$		instantiation
	|	$\exists x.\ A$		generation
	|	$!\ A$		replication

Generalizes FO multiset rewriting (MSR 1-2)

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

State and Transitions

- States

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

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

  - Σ is a list
  - Γ and Δ are commutative monoids

  - Constructor: ","
  - Empty: "•"

- Transitions

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

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

  - $\rightarrow^{*}$ for reflexive and transitive closure

Transition Semantics

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

$\top$ $\quad$ (no rule)

$\&$ $\quad \Sigma ; \Gamma ; (\Delta, A_1 \,\&\, A_2) \rightarrow \Sigma ; \Gamma ; (\Delta, A_i)$

$\forall$ $\quad \Sigma ; \Gamma ; (\Delta, \forall x.\, A) \rightarrow \Sigma ; \Gamma ; (\Delta, [t/x]A)$

if $\Sigma \vdash t$

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

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

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

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

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

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

Linear Logic

- Formulas

  $A, B ::= a \mid 1 \mid A \otimes B \mid A \multimap B \mid !A \mid \top \mid A \,\&\, B \mid \forall x.\, A \mid \exists x.\, A$

- LV sequents

  - Constructor: ","
  - Empty: "•"

$$\Gamma \,;\, \Delta \;\text{-->}_{\Sigma}\; C$$

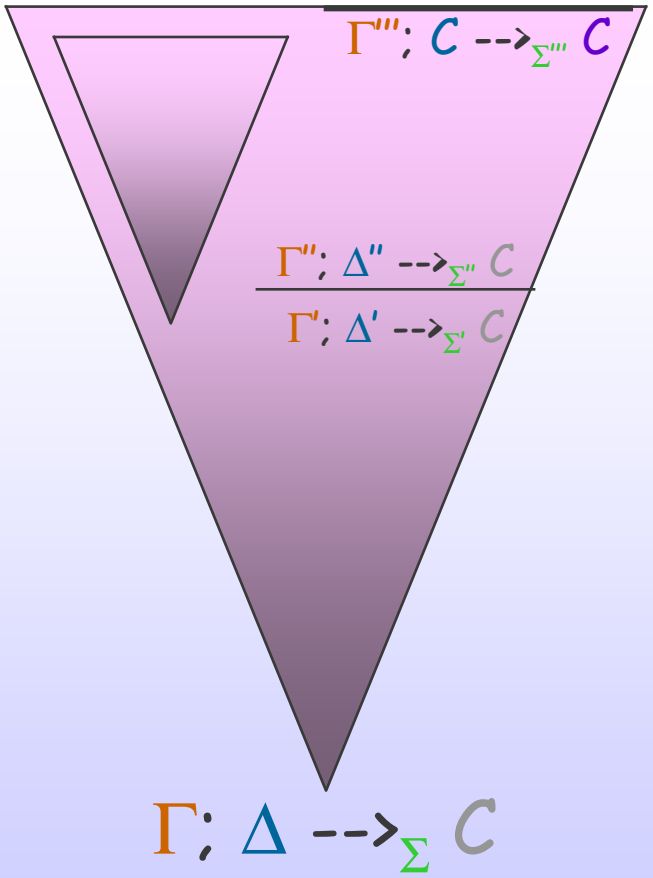
Unrestricted context

Linear context

Signature

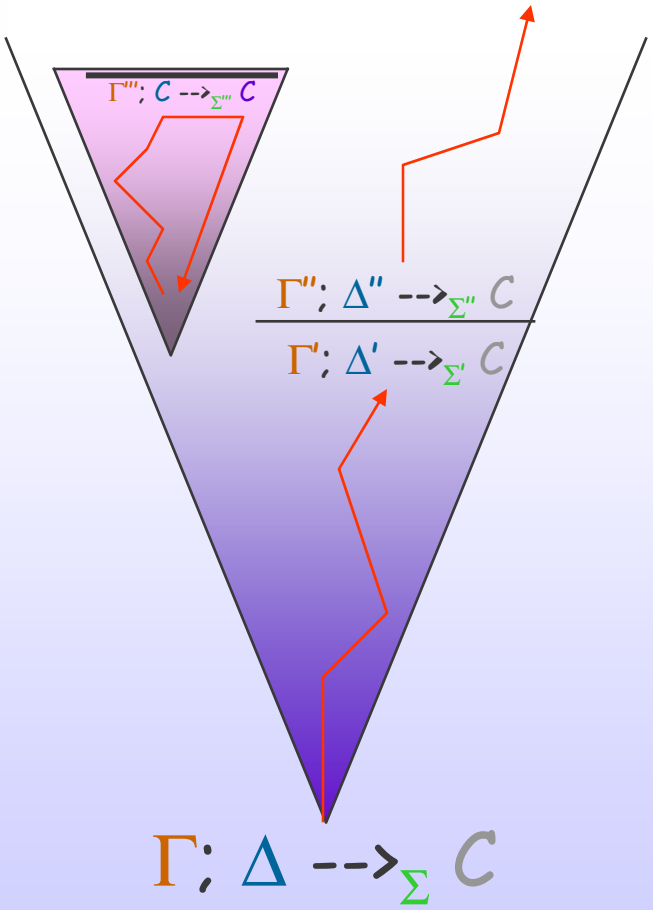
Goal formula

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

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

Interpreting Unary Rules

$$\frac{\Gamma;\ \Delta,\ A,\ B \dashrightarrow_{\Sigma} C}{\Gamma;\ \Delta,\ A \otimes B \dashrightarrow_{\Sigma} C} \qquad \Sigma;\ \Gamma;\ (\Delta,\ A \otimes B) \rightarrow \Sigma;\ \Gamma;\ (\Delta,\ A,\ B)$$

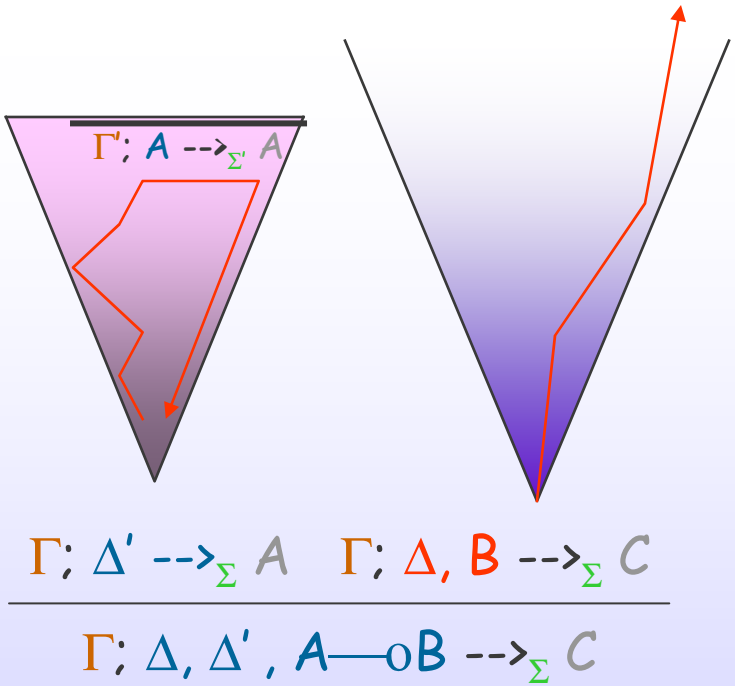
$$\frac{\Sigma \vdash t \quad \Gamma;\ \Delta,\ [t/x]A \dashrightarrow_{\Sigma} C}{\Gamma;\ \Delta,\ \forall x.A \dashrightarrow_{\Sigma} C} \qquad \Sigma;\ \Gamma;\ (\Delta,\ \forall x.\ A) \rightarrow \Sigma;\ \Gamma;\ (\Delta,\ [t/x]A) \text{ if } \Sigma \vdash t$$

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

$$\frac{\Gamma,\ A;\ \Delta \dashrightarrow_{\Sigma} C}{\Gamma;\ \Delta\ ,\ !A \dashrightarrow_{\Sigma} C} \qquad \Sigma;\ \Gamma;\ (\Delta,\ !A) \rightarrow \Sigma;\ (\Gamma,\ A);\ \Delta$$

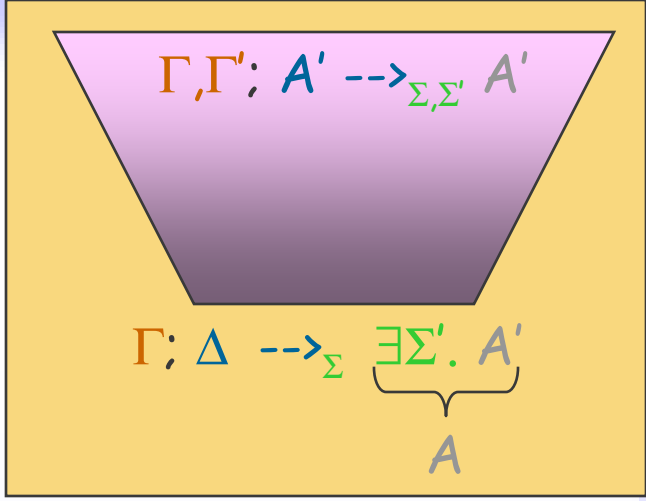
... ...

Binary Rules and Axiom



- Minor premise
  - Auxiliary rewrite chain
- Top of tree
  - Focus shifts to RHS
    - Axiom rule
  - Observation

Observations



- Observation states

$$\Sigma \quad ; \quad \Delta$$

  - 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

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

Interpreting Binary Rules

$$\frac{}{\Gamma;\ A \dashrightarrow_{\Sigma} A}$$

$\Sigma;\ \Gamma;\ \Delta \rightarrow^{*} \Sigma;\ \Delta$

$\Sigma;\ \Gamma;\ \Delta \rightarrow^{*} \Sigma'';\ \Delta''$
if $\Sigma;\ \Gamma;\ \Delta \rightarrow \Sigma';\ \Gamma';\ \Delta'$
and $\Sigma';\ \Gamma';\ \Delta' \rightarrow^{*} \Sigma'';\ \Delta''$

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

$\Sigma;\ \Gamma;\ (\Delta, \Delta', A \multimap B) \rightarrow \Sigma;\ \Gamma;\ (\Delta, B)$
if $\Sigma;\ \Gamma;\ \Delta' \rightarrow^{*} \Sigma;\ A$

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

$\Sigma;\ \Gamma;\ \Delta, \Delta' \rightarrow \Sigma;\ \Gamma;\ (A, \Delta)$
if $\Sigma;\ \Gamma;\ \Delta' \rightarrow^{*} \Sigma;\ A$

...

...

Formal Correspondence

- Soundness

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

  then $\Gamma ; \Delta \dashrightarrow_{\Sigma} \exists \Sigma'. \otimes \Delta'$

- Completeness?
  - No! We have only crippled right rules

    $\bullet ; \bullet ; a \multimap b, b \multimap c \not\rightarrow^* \bullet ; a \multimap c$

System ω

- With cut, rule for —o can be simplified to

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

- Cut elimination holds

  = in-lining of auxiliary rewrite chains

  - But …
    - Careful with extra signature symbols
    - Careful with extra persistent objects

- No rule for → needs a premise
  - → does not depend on →*

Multiset Rewriting

- Multiset: set with repetitions allowed

  $a ::= \bullet \mid a, a$

  - 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

The Atomic Objects of MSR 3

Atomic terms

- Principals — A
- Keys — K
- Nonces — N
- Other
  - Raw data, timestamp, …

Constructors

- Encryption — $\{\_\}_\_$
- Pairing — $(\_, \_)$
- Other
  - Signature, hash, MAC, …

Predicates

- Network — net
- Memory — M_A
- Intruder — I
- …

Fully definable

Types

- Simple types
  - A : princ
  - n : nonce
  - m : msg, …
- Dependent types
  - k : shK A B
  - K : pubK A
  - K' : privK K, …

Fully definable

- Powerful abstraction mechanism
  - At various user-definable level
    - Finely tagged messages
    - Untyped: msg only
- Simplify specification and reasoning
- Automated type checking

Subsorting

$$\tau <: \tau'$$

- Allows atomic terms in messages
- Definable
  - Non-transmittable terms
  - Sub-hierarchies
- Discriminant for type-flaw attacks

Data Access Specification

- Prevent illegitimate use of information
  - Protocol specification divided in roles
    - Owner = principal executing the role
  - A signing/encrypting with B's key
  - A accessing B's private data, …
- Simple static check
- Central meta-theoretic notion
  - Detailed specification of Dolev-Yao access model
- Gives meaning to Dolev-Yao intruder
- Current effort towards integration in type system
  - Definable
    - Possibility of going beyond Dolev-Yao model

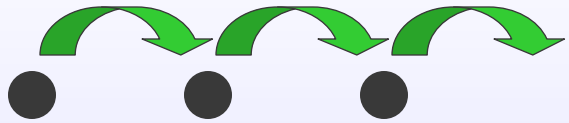
Modules and Equations

- Modules
  - Bundle declarations with simple import/export interface
  - Keep specifications tidy
  - Reusable
- Equations

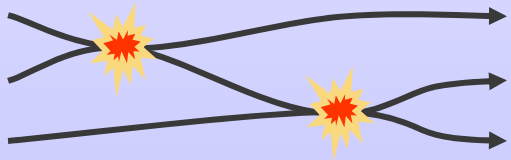
  (For free from underlying Maude engine)
  - Specify useful algebraic properties
    - Associativity of pairs
  - Allow to go beyond free-algebra model
    - Dec(k, Enc(k, M)) = M

State-Based vs. Process-Based

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

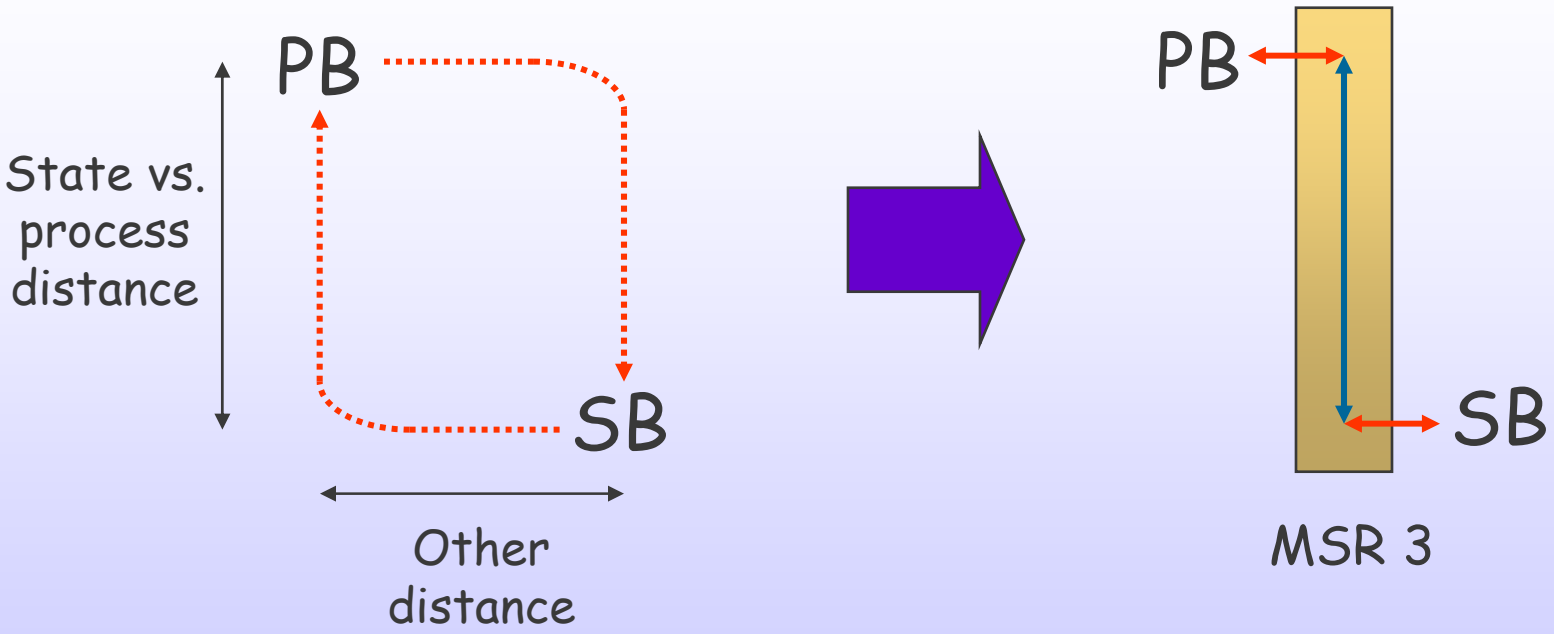


- Process-based languages
  - Process Algebra
  - Strand spaces, spi-calculus, …
  - ➢ Independent communicating threads



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

Summary

- MSR 3.0
  - Language for security protocol specification
  - Succinct representations
    - Simpl specifications
    - Economy of reasoning
  - Bridge between
    - State-based representation
    - Process-based representation
- 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