Part 4
Spatial Logics

Luca Cardelli
Andy Gordon | Luis Caires

Properties of Secure Mobile Computation

- We would like to express properties of unique, private, hidden, and secret *names*:
  - "The applet is placed in a private sandbox."
  - "The key exchange happens in a secret location."
  - "A shared private key is established between two locations."
  - "A fresh nonce is generated and transmitted."
- Crucial to expressing this kind of properties is devising new logical quantifiers for *fresh* and *hidden* entities:
  - "There is a fresh (never used before) name such that …"
  - "There is a hidden (unnamable) location such that …"
  - N.B.: standard quantifiers are problematic. "There exists a sandbox containing the applet" is rather different from "There exists a fresh sandbox containing the applet" and from "There exists a hidden sandbox containing the applet".

Approach

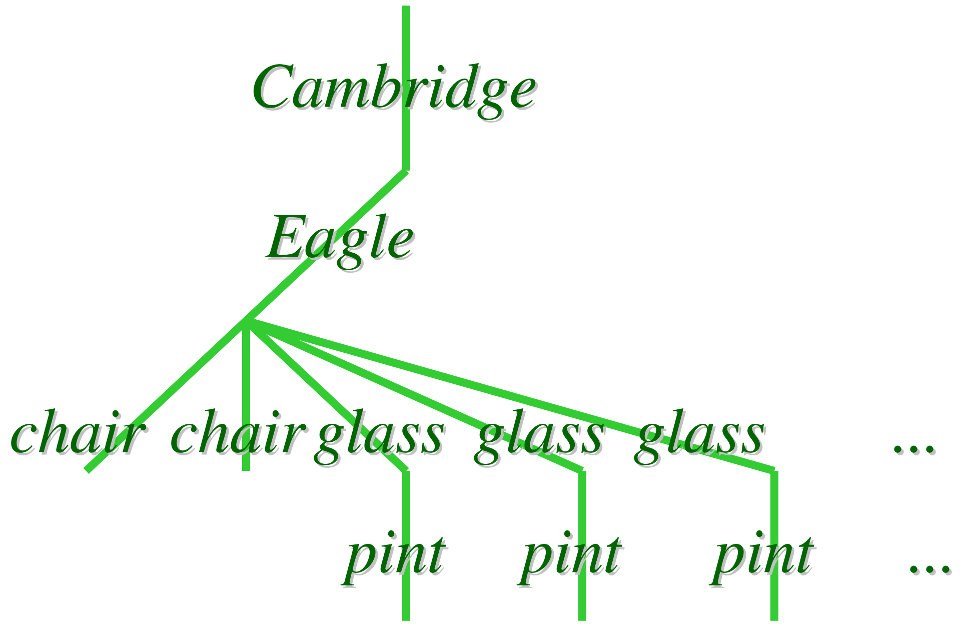
- Use a specification logic grounded in an operational model of mobility. (So soundness is not an issue.)
- Express properties of dynamically changing structures of locations.
  - Previous work [POPL'00].
- Express properties of hidden names. We split it into two logical tasks:
  - Quantify over fresh names. We adopt [Gabbay-Pitts].
  - Reveal hidden names, so we can talk about them.
  - Combine the two, to quantify over hidden locations.

    "There is a hidden location …" represented as:

    "There is a fresh name that can be used to reveal (mention) the hidden name of a location …".

Spatial Structures

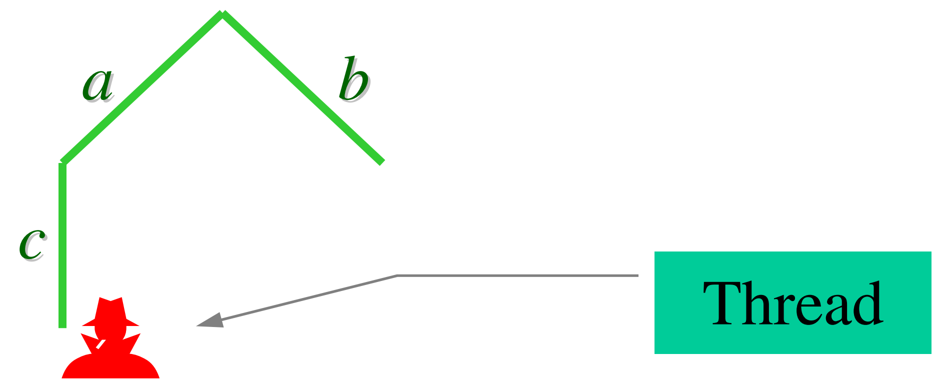
- Our basic model of space is going to be *finite-depth edge-labeled unordered trees* (*c.f.* semistructured data, XML). For short: *spatial trees*, represented by a syntax of *spatial expressions*. Unbounded resources are represented by infinite branching:



Cambridge[*Eagle*[*chair*[**0**] | *chair*[**0**] | !*glass*[*pint*[**0**]]] | ...]

Ambient Structures

- These spatial expressions/trees are a subset of ambient expressions/trees, which can represent both the spatial and the temporal aspects of mobile computation.



- An ambient tree is a spatial tree with, possibly, threads at each node that can locally change the shape of the tree.

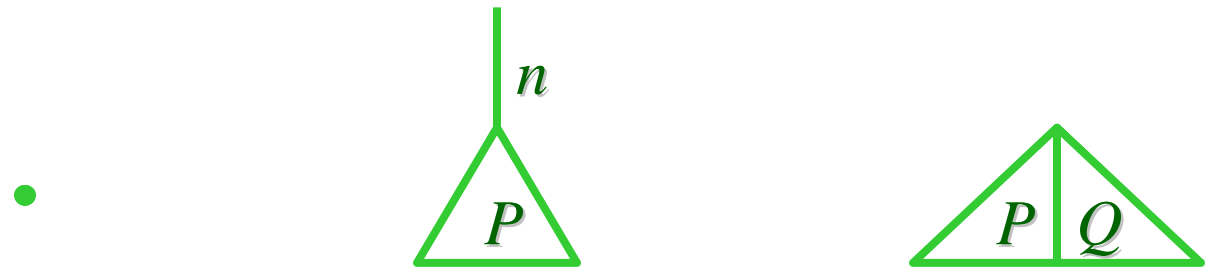
$$a[c[\mathit{out}\ a.\ \mathit{in}\ b.\ P]]\ |\ b[\mathbf{0}]$$

Spatial Logics

- We want to describe mobile behaviors. The *ambient calculus* provides an operational model, where spatial structures (agents, networks, etc.) are represented by nested locations.
- We also want to specify mobile behaviors. To this end, we devise an *ambient logic* that can talk about spatial structures.

Processes		*Formulas*	
$\mathbf{0}$	(void)	$\mathbf{0}$	(there is nothing here)
$n[P]$	(location)	$n[\mathcal{A}]$	(there is one thing here)
$P \mid Q$	(composition)	$\mathcal{A} \mid \mathcal{B}$	(there are two things here)

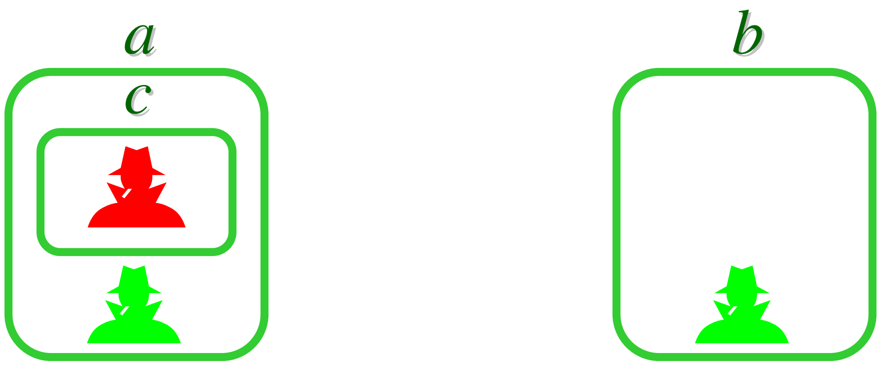
Trees

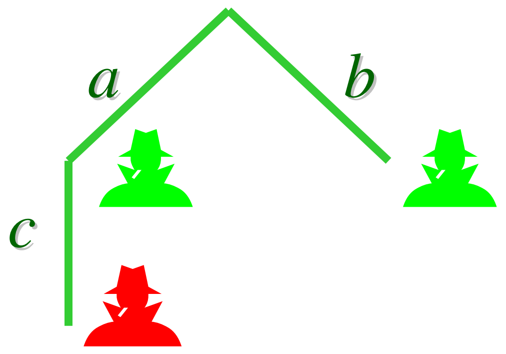


(void) (location) (composition)

Mobility

- *Mobility* is change of spatial structures over time.

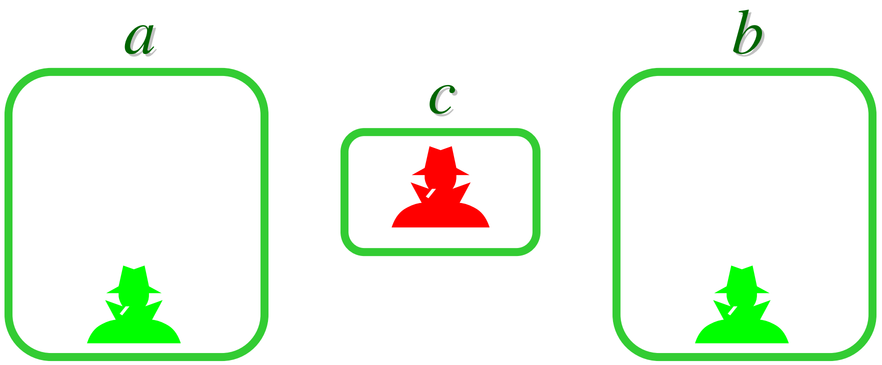


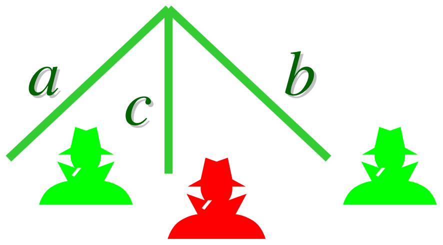


$a[Q \mid c[\mathit{out}\ a.\ \mathit{in}\ b.\ P]] \qquad \mid b[R]$

Mobility

- *Mobility* is change of spatial structures over time.

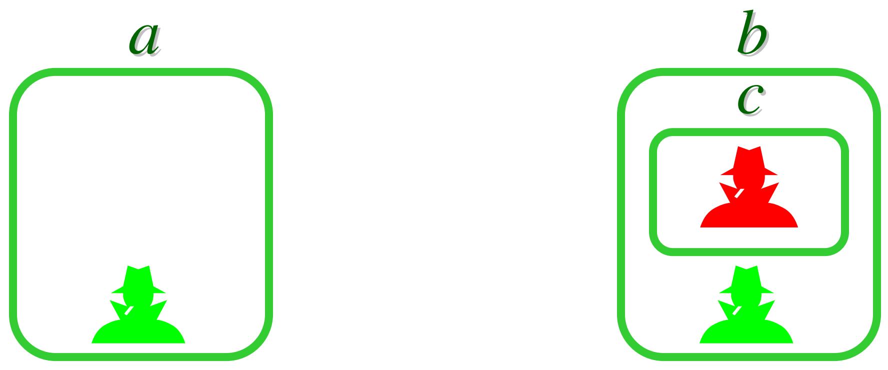


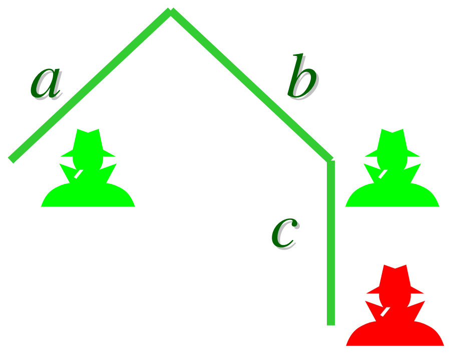


$a[Q] \quad | \; c[\mathit{in}\ b.\ P] \;\; | \; b[R]$

Mobility

- *Mobility* is change of spatial structures over time.

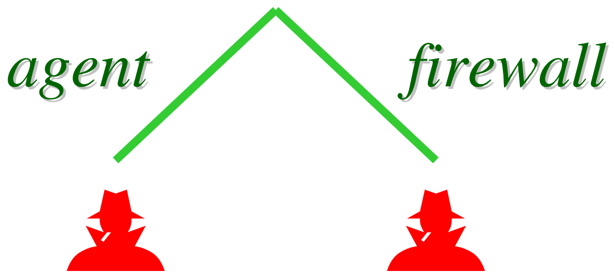




$a[Q] \mid b[R \mid c[P]]$

Properties of Mobile Computation

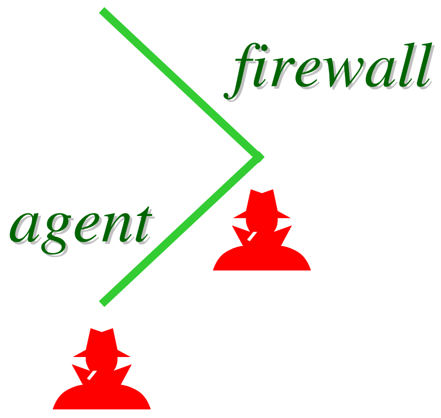
- These often have the form:
  - Right now, we have a spatial configuration, and later, we have another spatial configuration.
  - E.g.: Right now, the agent is outside the firewall, …



Now

Properties of Mobile Computation

- These often have the form:
  - Right now, we have a spatial configuration, and later, we have another spatial configuration.
  - E.g.: Right now, the agent is outside the firewall, and later (after running an authentication protocol), the agent is inside the firewall.



Later

Modal Logics

- In a modal logic, the truth of a formula is relative to a state (called a *world*).
  - Temporal logic: current time.
  - Program logic: current store contents.
  - Epistemic logic: current knowledge. Etc.
- In our case, the truth of a *space-time modal formula* is relative to the *here and now* of a process.
  - The formula $n[\mathbf{0}]$ is read:

    there is *here and now* an empty location called n
  - The operator $n[\mathcal{A}]$ is a single step in space (akin to the temporal next), which allows us talk about that place one step down into n.
  - Other modal operators talk about undetermined times (in the future) and undetermined places (in the location tree).

Logical Formulas

$\mathcal{A} \in \Phi ::=$	Formulas	(η is a name n or a variable x)	
$\mathbf{T}$	true		
$\neg\mathcal{A}$	negation		
$\mathcal{A} \vee \mathcal{A}'$	disjunction		
$\mathbf{0}$	void		
$\eta[\mathcal{A}]$	location	$\mathcal{A}@\eta$	location adjunct
$\mathcal{A} \mid \mathcal{A}'$	composition	$\mathcal{A} \triangleright \mathcal{A}'$	composition adjunct
$\eta \circledR \mathcal{A}$	revelation	$\mathcal{A} \oslash \eta$	revelation adjunct
$✧\mathcal{A}$	somewhere modality		
$\Diamond\mathcal{A}$	sometime modality		
$\forall x.\mathcal{A}$	universal quantification over names		

Simple Examples

❶: $p[\mathbf{T}] \mid \mathbf{T}$

there is a location p here (and possibly something else)

❷: $\diamond$❶

somewhere there is a location p

❸: ❷ $\Rightarrow \Box$❷

if there is a p somewhere, then forever there is a p somewhere

❹: $p[q[\mathbf{T}] \mid \mathbf{T}] \mid \mathbf{T}$

there is a p with a child q here

❺: $\diamond$❹

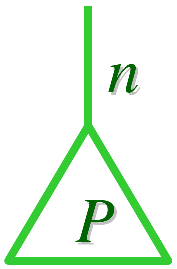
somewhere there is a p with a child q

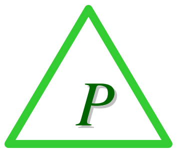
Examples

• $an\ n \triangleq n[\mathbf{T}] \mid \mathbf{T}$	there is now an n here
• $no\ n \triangleq \neg an\ n$	there is now no n here
• $one\ n \triangleq n[\mathbf{T}] \mid no\ n$	there is now exactly one n here
• $\mathcal{A}^{\forall} \triangleq \neg(\neg\mathcal{A} \mid \mathbf{T})$	everybody here satisfies $\mathcal{A}$
• $(n[\mathbf{T}] \Rightarrow n[\mathcal{A}])^{\forall}$	every n here satisfies $\mathcal{A}$
• ¤$((n[\mathbf{T}] \Rightarrow n[\mathcal{A}])^{\forall})$	every n everywhere satisfies $\mathcal{A}$

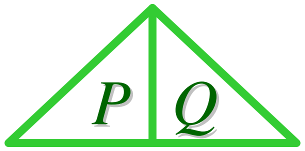
Satisfaction for Basic Trees

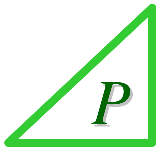
- $\models \mathbf{0}$

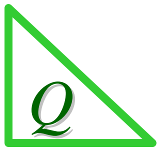


$\models n[\mathcal{A}]$ if 

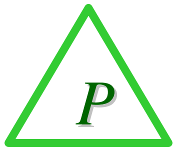
$\models \mathcal{A}$

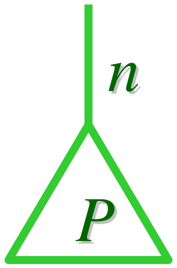


$\models \mathcal{A} \mid \mathcal{B}$ if 

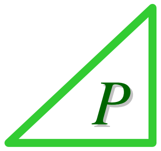
$\models \mathcal{A}$ and 

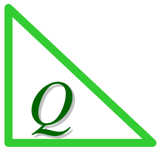
$\models \mathcal{B}$

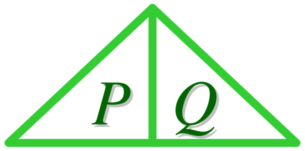


$\models \mathcal{A}@n$ if 

$\models \mathcal{A}$

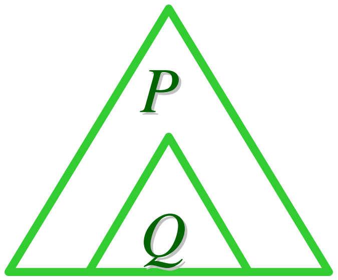


$\models \mathcal{A} \triangleright \mathcal{B}$ if for all 

$\models \mathcal{A}$ we have 

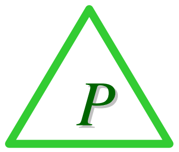
$\models \mathcal{B}$

Satisfaction for Somewhere/Sometime

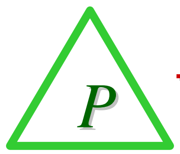


$\models \lozenge\!\!\!\!\!\diamond\,\mathcal{A}$ if

$\models \mathcal{A}$



$\models \Diamond\mathcal{A}$ if



$\longrightarrow^{*}$

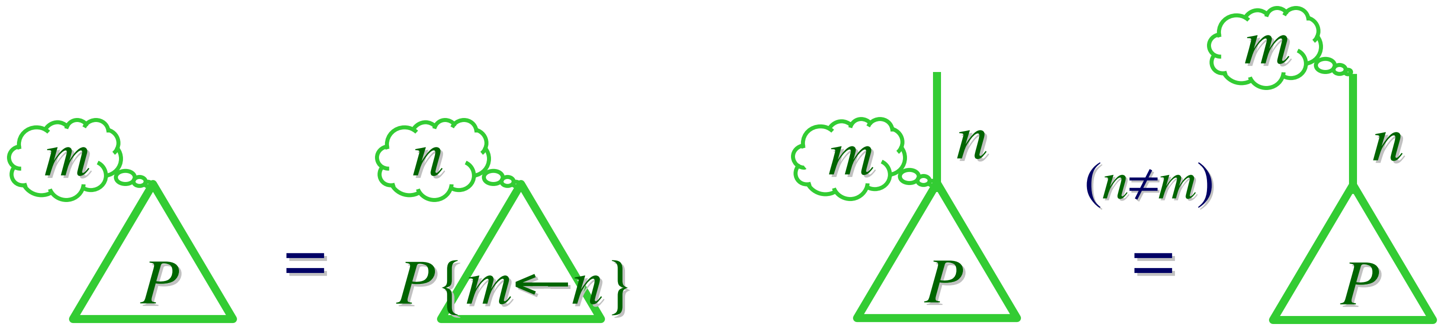
and

$\models \mathcal{A}$

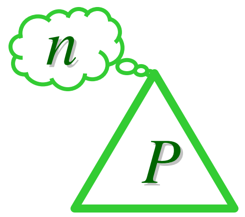
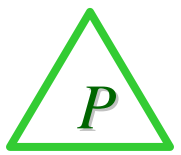
- N.B.: instead of $\Diamond\mathcal{A}$ and ✧$\mathcal{A}$ we can use a “temporal next” operator $\circ\mathcal{A}$, along with the existing “spatial next” operator $n[\mathcal{A}]$, together with μ-calculus style recursive formulas.

Satisfaction for Revelation

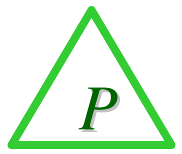
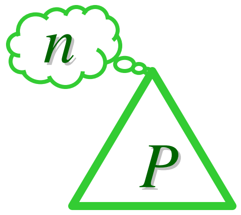
- Trees with hidden labels:



Etc.

 $\vDash n \circledR \mathcal{A}$ if  $\vDash \mathcal{A}$

Not possible if *n* is free!

 $\vDash \mathcal{A} \oslash n$ if  $\vDash \mathcal{A}$

Intended Model: Ambient Calculus

$P \in \Pi ::=$	Processes		$M ::=$	Messages
$(\nu n)P$	restriction	Location Trees	n	name
$\mathbf{0}$	inactivity		*in* M	entry capability
$P \mid P'$	parallel		*out* M	exit capability
$M[P]$	ambient		*open* M	open capability
$!P$	replication		ε	empty path
$M.P$	exercise a capability	Actions	$M.M'$	composite path
$(n).P$	input locally, bind to n			
$\langle M \rangle$	output locally (async)			

$n[] \quad \triangleq \quad n[\mathbf{0}]$

$M \quad \triangleq \quad M.\mathbf{0}$ (where appropriate)

Reduction Semantics

- A structural congruence relation $P \equiv Q$:
  - On spatial expressions, $P \equiv Q$ iff P and Q denote the same tree. So, the syntax modulo $\equiv$ is a notation for spatial trees.
  - On full ambient expressions, $P \equiv Q$ if in addition the respective threads are "trivially equivalent".
  - Prominent in the definition of the logic.
- A reduction relation $P \longrightarrow^* Q$:
  - Defining the meaning of mobility and communication actions.
  - Closed up to structural congruence:

$$P \equiv P', \ P' \longrightarrow^* Q', \ Q' \equiv Q \quad \Rightarrow \quad P \longrightarrow^* Q$$

Reduction

- Four basic reductions plus propagation, rearrangement (composition with structural congruence), and transitivity.

$n[in\ m.\ P \mid Q] \mid m[R]$	$\longrightarrow m[n[P \mid Q] \mid R]$	(Red In)
$m[n[out\ m.\ P \mid Q] \mid R]$	$\longrightarrow n[P \mid Q] \mid m[R]$	(Red Out)
$open\ m.\ P \mid m[Q]$	$\longrightarrow P \mid Q$	(Red Open)
$(n).P \mid \langle M \rangle$	$\longrightarrow P\{n \leftarrow M\}$	(Red Comm)
$P \longrightarrow Q \ \Rightarrow\ (\nu n)P \longrightarrow (\nu n)Q$		(Red Res)
$P \longrightarrow Q \ \Rightarrow\ n[P] \longrightarrow n[Q]$		(Red Amb)
$P \longrightarrow Q \ \Rightarrow\ P \mid R \longrightarrow Q \mid R$		(Red Par)
$P' \equiv P,\ P \longrightarrow Q,\ Q \equiv Q' \ \Rightarrow\ P' \longrightarrow Q'$		(Red $\equiv$)

$\longrightarrow^*$ is the reflexive-transitive closure of $\longrightarrow$

Structural Congruence

- Routine, but used heavily in the logic and semantics.

$P \equiv P$	(Struct Refl)
$P \equiv Q \;\Rightarrow\; Q \equiv P$	(Struct Symm)
$P \equiv Q, Q \equiv R \;\Rightarrow\; P \equiv R$	(Struct Trans)
$P \equiv Q \;\Rightarrow\; (\nu n)P \equiv (\nu n)Q$	(Struct Res)
$P \equiv Q \;\Rightarrow\; P \mid R \equiv Q \mid R$	(Struct Par)
$P \equiv Q \;\Rightarrow\; !P \equiv !Q$	(Struct Repl)
$P \equiv Q \;\Rightarrow\; M[P] \equiv M[Q]$	(Struct Amb)
$P \equiv Q \;\Rightarrow\; M.P \equiv M.Q$	(Struct Action)
$P \equiv Q \;\Rightarrow\; (n).P \equiv (n).Q$	(Struct Input)
$\varepsilon.P \equiv P$	(Struct ε)
$(M.M').P \equiv M.M'.P$	(Struct .)

$(\nu n)\mathbf{0} \equiv \mathbf{0}$	(Struct Res Zero)
$(\nu n)(\nu m)P \equiv (\nu m)(\nu n)P$	(Struct Res Res)
$(\nu n)(P \mid Q) \equiv P \mid (\nu n)Q \quad \text{if } n \notin fn(P)$	(Struct Res Par)
$(\nu n)(m[P]) \equiv m[(\nu n)P] \quad \text{if } n \neq m$	(Struct Res Amb)
$P \mid Q \equiv Q \mid P$	(Struct Par Comm)
$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$	(Struct Par Assoc)
$P \mid \mathbf{0} \equiv P$	(Struct Par Zero)
$!(P \mid Q) \equiv !P \mid !Q$	(Struct Repl Par)
$!\mathbf{0} \equiv \mathbf{0}$	(Struct Repl Zero)
$!P \equiv P \mid !P$	(Struct Repl Copy)
$!P \equiv !!P$	(Struct Repl Repl)

- These axioms (particularly the ones for **!**) are sound and complete with respect to equality of spatial trees: edge-labeled finite-depth unordered trees, with infinite-branching but finitely many distinct labels under each node.

Satisfaction: Basic Tree Formulas

$P \vDash \mathbf{0}$	$\triangleq$	$P \equiv \mathbf{0}$
$P \vDash n[\mathcal{A}]$	$\triangleq$	$\exists P' \in \Pi.\ P \equiv n[P'] \wedge P' \vDash \mathcal{A}$
$P \vDash \mathcal{A} \mid \mathcal{B}$	$\triangleq$	$\exists P', P'' \in \Pi.\ P \equiv P' \mid P'' \wedge P' \vDash \mathcal{A} \wedge P'' \vDash \mathcal{B}$
$P \vDash \mathcal{A}@n$	$\triangleq$	$n[P] \vDash \mathcal{A}$
$P \vDash \mathcal{A} \triangleright \mathcal{B}$	$\triangleq$	$\forall P' \in \Pi.\ P' \vDash \mathcal{A} \Rightarrow P \mid P' \vDash \mathcal{B}$

- $\mathbf{0}$: there is no structure here now.
- $n[\mathcal{A}]$: there is a location n with contents satisfying $\mathcal{A}$.
- $\mathcal{A} \mid \mathcal{B}$: there are two structures satisfying $\mathcal{A}$ and $\mathcal{B}$.
- $\mathcal{A}@n$: when the current structure is placed in a location n, the resulting structure satisfies $\mathcal{A}$.
- $\mathcal{A} \triangleright \mathcal{B}$: when the current structure is composed with one satisfying $\mathcal{A}$, the resulting structures satisfies $\mathcal{B}$.

Meaning of Formulas: Satisfaction Relation

$P \models \mathbf{T}$		
$P \models \neg\mathcal{A}$	$\triangleq$	$\neg\, P \models \mathcal{A}$
$P \models \mathcal{A} \vee \mathcal{B}$	$\triangleq$	$P \models \mathcal{A} \vee P \models \mathcal{B}$
$P \models \mathbf{0}$	$\triangleq$	$P \equiv \mathbf{0}$
$P \models n[\mathcal{A}]$	$\triangleq$	$\exists P' \in \Pi.\ P \equiv n[P'] \wedge P' \models \mathcal{A}$
$P \models \mathcal{A}@n$	$\triangleq$	$n[P] \models \mathcal{A}$
$P \models \mathcal{A} \mid \mathcal{B}$	$\triangleq$	$\exists P', P'' \in \Pi.\ P \equiv P' \mid P'' \wedge P' \models \mathcal{A} \wedge P'' \models \mathcal{B}$
$P \models \mathcal{A} \triangleright \mathcal{B}$	$\triangleq$	$\forall P' \in \Pi.\ P' \models \mathcal{A} \Rightarrow P \mid P' \models \mathcal{B}$
$P \models n \circledR \mathcal{A}$	$\triangleq$	$\exists P' \in \Pi.\ P \equiv (\nu n)P' \wedge P' \models \mathcal{A}$
$P \models \mathcal{A} \oslash n$	$\triangleq$	$(\nu n)P \models \mathcal{A}$
$P \models \text{✧}\mathcal{A}$	$\triangleq$	$\exists P' \in \Pi.\ P \downarrow^{*} P' \wedge P' \models \mathcal{A}$
$P \models \Diamond\mathcal{A}$	$\triangleq$	$\exists P' \in \Pi.\ P \longrightarrow^{*} P' \wedge P' \models \mathcal{A}$
$P \models \forall x.\mathcal{A}$	$\triangleq$	$\forall m \in \Lambda.\ P \models \mathcal{A}\{x \leftarrow m\}$

$P \downarrow P'$ iff $\exists n, P''.\ P \equiv n[P'] \mid P''$; $\downarrow^{*}$ is the refl-trans closure of $\downarrow$

Basic Fact

- Satisfaction is invariant under structural congruence:

$$P \vDash \mathcal{A},\ P \equiv P' \quad \Rightarrow \quad P' \vDash \mathcal{A}$$

  I.e.: $\{P \in \Pi \mid P \vDash \mathcal{A}\}$ is closed under $\equiv$.

- Hence, formulas describe congruence-invariant properties.
  - In particular, formulas describe properties of spatial trees.
  - N.B.: Most process logics describe bisimulation-invariant properties.
- Hence, formulas talk about *trees*.

From Satisfaction to (Propositional) Logic

- Propositional validity

  $\mathbf{vld}\,\mathcal{A} \;\triangleq\; \forall P \in \Pi.\ P \vDash \mathcal{A}$ $\qquad$ $\mathcal{A}$ (closed) is valid

- Sequents

  $\mathcal{A} \vdash \mathcal{B} \;\triangleq\; \forall P \in \Pi.\ P \vDash \mathcal{A} \Rightarrow P \vDash \mathcal{B}$

- Rules

  $\mathcal{A}_1 \vdash \mathcal{B}_1;\ \ldots;\ \mathcal{A}_n \vdash \mathcal{B}_n \;\}\; \mathcal{A} \vdash \mathcal{B} \;\triangleq$ $\qquad$ $(n \geq 0)$

  $\mathcal{A}_1 \vdash \mathcal{B}_1 \wedge \ldots \wedge \mathcal{A}_n \vdash \mathcal{B}_n \Rightarrow \mathcal{A} \vdash \mathcal{B}$

  (N.B.: all the rules shown later are validated accordingly.)

- Conventions:
  - $\dashv\vdash$ means $\vdash$ in both directions

    $\{\}$ means $\}$ in both directions

Obtaining…

- Logical axioms and rules.
  - Rules of propositional logic (standard).
  - Rules of location and composition

    $\mathcal{A} \mid \mathcal{C} \vdash \mathcal{B} \;\{\}\; \mathcal{A} \vdash \mathcal{C} \triangleright \mathcal{B}$ | $\mid$-$\triangleright$ adjunction
  - Rules of revelation

    $\eta \circledR \mathcal{A} \vdash \mathcal{B} \;\{\}\; \mathcal{A} \vdash \mathcal{B} \oslash \eta$ ®-$\oslash$ adjunction

    $\}\; (\neg\mathcal{A}) \oslash x \dashv\vdash \neg(\mathcal{A} \oslash x)$ $\oslash$ is self-dual
  - Rules of ✧ and ◇ modalities (standard S4, plus some)
  - Rules of quantification (standard, but for name quantifiers)
- A large collection of logical consequences.

Rules: Propositional Calculus

(A-L)	$\mathcal{A}\wedge(\mathcal{C}\wedge\mathcal{D}) \vdash \mathcal{B} \ \{\} \ (\mathcal{A}\wedge\mathcal{C})\wedge\mathcal{D} \vdash \mathcal{B}$
(A-R)	$\mathcal{A} \vdash (\mathcal{C}\vee\mathcal{D})\vee\mathcal{B} \ \{\} \ \mathcal{A} \vdash \mathcal{C}\vee(\mathcal{D}\vee\mathcal{B})$
(X-L)	$\mathcal{A}\wedge\mathcal{C} \vdash \mathcal{B} \ \} \ \mathcal{C}\wedge\mathcal{A} \vdash \mathcal{B}$
(X-R)	$\mathcal{A} \vdash \mathcal{C}\vee\mathcal{B} \ \} \ \mathcal{A} \vdash \mathcal{B}\vee\mathcal{C}$
(C-L)	$\mathcal{A}\wedge\mathcal{A} \vdash \mathcal{B} \ \} \ \mathcal{A} \vdash \mathcal{B}$
(C-R)	$\mathcal{A} \vdash \mathcal{B}\vee\mathcal{B} \ \} \ \mathcal{A} \vdash \mathcal{B}$
(W-L)	$\mathcal{A} \vdash \mathcal{B} \ \} \ \mathcal{A}\wedge\mathcal{C} \vdash \mathcal{B}$
(W-R)	$\mathcal{A} \vdash \mathcal{B} \ \} \ \mathcal{A} \vdash \mathcal{C}\vee\mathcal{B}$
(Id)	$\} \ \mathcal{A} \vdash \mathcal{A}$
(Cut)	$\mathcal{A} \vdash \mathcal{C}\vee\mathcal{B}; \ \mathcal{A}'\wedge\mathcal{C} \vdash \mathcal{B}' \ \} \ \mathcal{A}\wedge\mathcal{A}' \vdash \mathcal{B}\vee\mathcal{B}'$
(**T**)	$\mathcal{A}\wedge\mathbf{T} \vdash \mathcal{B} \ \} \ \mathcal{A} \vdash \mathcal{B}$
(**F**)	$\mathcal{A} \vdash \mathbf{F}\vee\mathcal{B} \ \} \ \mathcal{A} \vdash \mathcal{B}$
(¬-L)	$\mathcal{A} \vdash \mathcal{C}\vee\mathcal{B} \ \} \ \mathcal{A}\wedge\neg\mathcal{C} \vdash \mathcal{B}$
(¬-R)	$\mathcal{A}\wedge\mathcal{C} \vdash \mathcal{B} \ \} \ \mathcal{A} \vdash \neg\mathcal{C}\vee\mathcal{B}$

Rules: Composition

$(\mid \mathbf{0})$	$\} \; \mathcal{A} \mid \mathbf{0} \dashv\vdash \mathcal{A}$	$\mathbf{0}$ is nothing
$(\mid \neg\mathbf{0})$	$\} \; \mathcal{A} \mid \neg\mathbf{0} \vdash \neg\mathbf{0}$	if a part is non-$\mathbf{0}$, so is the whole
$(\mathbf{A} \mid)$	$\} \; \mathcal{A} \mid (\mathcal{B} \mid \mathcal{C}) \dashv\vdash (\mathcal{A} \mid \mathcal{B}) \mid \mathcal{C}$	$\mid$ associativity
$(\mathbf{X} \mid)$	$\} \; \mathcal{A} \mid \mathcal{B} \vdash \mathcal{B} \mid \mathcal{A}$	$\mid$ commutativity
$(\mid \vdash)$	$\mathcal{A}' \vdash \mathcal{B}'; \; \mathcal{A}'' \vdash \mathcal{B}'' \; \} \; \mathcal{A}' \mid \mathcal{A}'' \vdash \mathcal{B}' \mid \mathcal{B}''$	$\mid$ congruence
$(\mid \vee)$	$\} \; (\mathcal{A} \vee \mathcal{B}) \mid \mathcal{C} \vdash \mathcal{A} \mid \mathcal{C} \vee \mathcal{B} \mid \mathcal{C}$	$\mid$-$\vee$ distribution
$(\mid \Vert)$	$\} \; \mathcal{A}' \mid \mathcal{A}'' \vdash \mathcal{A}' \mid \mathcal{B}'' \vee \mathcal{B}' \mid \mathcal{A}'' \vee \neg\mathcal{B}' \mid \neg\mathcal{B}''$	decomposition
$(\mid \triangleright)$	$\mathcal{A} \mid \mathcal{C} \vdash \mathcal{B} \; \{\} \; \mathcal{A} \vdash \mathcal{C} \triangleright \mathcal{B}$	$\mid$-$\triangleright$ adjunction
$(\triangleright\mathbf{F} \neg)$	$\} \; \mathcal{A}^{\mathbf{F}} \vdash \mathcal{A}^{\neg}$	if $\mathcal{A}$ is unsatisfiable then $\mathcal{A}$ is false
$(\neg \triangleright\mathbf{F})$	$\} \; \mathcal{A}^{\mathbf{F}\neg} \vdash \mathcal{A}^{\mathbf{FF}}$	if $\mathcal{A}$ is satisfiable then $\mathcal{A}^{\mathbf{F}}$ is unsatisfiable

where $\mathcal{A}^{\neg} \triangleq \neg\mathcal{A}$ and $\mathcal{A}^{\mathbf{F}} \triangleq \mathcal{A} \triangleright \mathbf{F}$

The Composition Adjunct

$(\mid\triangleright)\quad \mathcal{A} \mid \mathcal{C} \vdash \mathcal{B} \;\{\}\; \mathcal{A} \vdash \mathcal{C} \triangleright \mathcal{B}$

"Assume that every process that has a partition into pieces that satisfy $\mathcal{A}$ and $\mathcal{C}$, also satisfies $\mathcal{B}$. Then, every process that satisfies $\mathcal{A}$, together with any process that satisfies $\mathcal{C}$, satisfies $\mathcal{B}$. (And vice versa.)" (*c.f.* ($\multimap$ R))

- Interpretations of $\mathcal{A} \triangleright \mathcal{B}$:
  - P provides $\mathcal{B}$ in any context that provides $\mathcal{A}$
  - P ensures $\mathcal{B}$ under any attack that ensures $\mathcal{A}$

That is, $P \models \mathcal{A} \triangleright \mathcal{B}$ is a context-system spec (a concurrent version of a pre-post spec).

Moreover $\mathcal{A} \triangleright \mathcal{B}$ is, in a precise sense, linear implication: the context that satisfies $\mathcal{A}$ is used exactly once in the system that satisfies $\mathcal{B}$.

Some Derived Rules

$\} \ (\mathcal{A} \rhd \mathcal{B}) \mid \mathcal{A} \vdash \mathcal{B}$

"If P provides $\mathcal{B}$ in any context that provides $\mathcal{A}$, and Q provides $\mathcal{A}$, then P and Q together provide $\mathcal{B}$."

- Proof: $\mathcal{A} \rhd \mathcal{B} \vdash \mathcal{A} \rhd \mathcal{B} \ \} \ (\mathcal{A} \rhd \mathcal{B}) \mid \mathcal{A} \vdash \mathcal{B}$ by (Id), ($\mid \rhd$)

$\mathcal{D} \vdash \mathcal{A};\ \mathcal{B} \vdash \mathcal{C} \ \} \ \mathcal{D} \mid (\mathcal{A} \rhd \mathcal{B}) \vdash \mathcal{C}$ **(*c.f.* ($\multimap$ L))**

"If anything that satisfies $\mathcal{D}$ satisfies $\mathcal{A}$, and anything that satisfies $\mathcal{B}$ satisfies $\mathcal{C}$, then: anything that has a partition into a piece satisfying $\mathcal{D}$ (and hence $\mathcal{A}$), and another piece satisfying $\mathcal{B}$ in a context that satisfies $\mathcal{A}$, it satisfies ($\mathcal{B}$ and hence) $\mathcal{C}$."

- Proof:

$\mathcal{D} \vdash \mathcal{A};\ \mathcal{A} \rhd \mathcal{B} \vdash \mathcal{A} \rhd \mathcal{B} \ \} \ \mathcal{D} \mid \mathcal{A} \rhd \mathcal{B} \vdash \mathcal{A} \mid \mathcal{A} \rhd \mathcal{B}$	assumption, (Id), ($\mid \vdash$)
$\mathcal{A} \mid \mathcal{A} \rhd \mathcal{B} \vdash \mathcal{B}$	above
$\mathcal{B} \vdash \mathcal{C}$	assumption

More Derived Rules

$\} \ \mathcal{A} \vdash \mathbf{T} \mid \mathcal{A}$	you can always add more pieces (if they are **0**)
$\} \ \mathbf{F} \mid \mathcal{A} \vdash \mathbf{F}$	if a piece is absurd, so is the whole
$\} \ \mathbf{0} \vdash \neg(\neg\mathbf{0} \mid \neg\mathbf{0})$	**0** is single-threaded
$\} \ \mathcal{A} \mid \mathcal{B} \wedge \mathbf{0} \vdash \mathcal{A}$	you can split **0** (but you get **0**). Proof uses ($\mid$ $\Vert$)
$\mathcal{A}' \vdash \mathcal{A};\ \mathcal{B} \vdash \mathcal{B}' \ \} \ \mathcal{A}\triangleright\mathcal{B} \vdash \mathcal{A}'\triangleright\mathcal{B}'$	$\triangleright$ is contravariant on the left
$\} \ \mathcal{A}\triangleright\mathcal{B} \mid \mathcal{B}\triangleright\mathcal{C} \vdash \mathcal{A}\triangleright\mathcal{C}$	$\triangleright$ is transitive
$\} \ (\mathcal{A} \mid \mathcal{B})\triangleright\mathcal{C} \dashv\vdash \mathcal{A}\triangleright(\mathcal{B}\triangleright\mathcal{C})$	$\triangleright$ curry/uncurry
$\} \ \mathcal{A}\triangleright(\mathcal{B}\triangleright\mathcal{C}) \vdash \mathcal{B}\triangleright(\mathcal{A}\triangleright\mathcal{C})$	contexts commute
$\} \ \mathbf{T} \dashv\vdash \mathbf{T}\triangleright\mathbf{T}$	truth can withstand any attack
$\} \ \mathbf{T} \vdash \mathbf{F}\triangleright\mathcal{A}$	anything goes if you can find an absurd partner
$\} \ \mathbf{T}\triangleright\mathcal{A} \vdash \mathcal{A}$	if $\mathcal{A}$ resists any attack, then it holds

Rules: Location

$(n[]\ \neg 0)$	$\}\ n[\mathcal{A}] \vdash \neg 0$	locations exist
$(n[]\ \neg \mid)$	$\}\ n[\mathcal{A}] \vdash \neg(\neg 0 \mid \neg 0)$	are not decomposable
$(n[] \vdash)$	$\mathcal{A} \vdash \mathcal{B}\ \{\}\ n[\mathcal{A}] \vdash n[\mathcal{B}]$	$n[]$ congruence
$(n[]\ \wedge)$	$\}\ n[\mathcal{A}] \wedge n[\mathcal{C}] \vdash n[\mathcal{A} \wedge \mathcal{C}]$	$n[]$-$\wedge$ distribution
$(n[]\ \vee)$	$\}\ n[\mathcal{C} \vee \mathcal{B}] \vdash n[\mathcal{C}] \vee n[\mathcal{B}]$	$n[]$-$\vee$ distribution
$(n[]\ @)$	$n[\mathcal{A}] \vdash \mathcal{B}\ \{\}\ \mathcal{A} \vdash \mathcal{B}@n$	$n[]$-@ adjunction
$(\neg\ @)$	$\}\ \mathcal{A}@n \dashv\vdash \neg((\neg\mathcal{A})@n)$	@ is self-dual

Some Derived Rules

$\mathcal{A} \vdash \mathcal{B} \ \} \ \mathcal{A}@n \vdash \mathcal{B}@n$ @ congruence

$\} \ n[\mathcal{A}@n] \vdash \mathcal{A}$

$\} \ \mathcal{A} \dashv\vdash n[\mathcal{A}]@n$

$\} \ n[\neg\mathcal{A}] \vdash \neg n[\mathcal{A}]$

$\} \ \neg n[\mathcal{A}] \dashv\vdash \neg n[\mathbf{T}] \vee n[\neg\mathcal{A}]$

Rules: Time and Space Modalities

(◇)	$\}\ \Diamond\mathcal{A} \dashv\vdash \neg\Box\neg\mathcal{A}$	(✧)	$\}\ $✧$\mathcal{A} \dashv\vdash \neg$¤$\neg\mathcal{A}$
(□ K)	$\}\ \Box(\mathcal{A}\Rightarrow\mathcal{B}) \vdash \Box\mathcal{A}\Rightarrow\Box\mathcal{B}$	(¤ K)	$\}\ $¤$(\mathcal{A}\Rightarrow\mathcal{B}) \vdash $¤$\mathcal{A}\Rightarrow$¤$\mathcal{B}$
(□ T)	$\}\ \Box\mathcal{A} \vdash \mathcal{A}$	(¤ T)	$\}\ $¤$\mathcal{A} \vdash \mathcal{A}$
(□ 4)	$\}\ \Box\mathcal{A} \vdash \Box\Box\mathcal{A}$	(¤ 4)	$\}\ $¤$\mathcal{A} \vdash $¤¤$\mathcal{A}$
(□ **T**)	$\}\ \mathbf{T} \vdash \Box\mathbf{T}$	(¤ **T**)	$\}\ \mathbf{T} \vdash $¤$\mathbf{T}$
(□ ⊢)	$\mathcal{A} \vdash \mathcal{B}\ \}\ \Box\mathcal{A} \vdash \Box\mathcal{B}$	(¤ ⊢)	$\mathcal{A} \vdash \mathcal{B}\ \}\ $¤$\mathcal{A} \vdash $¤$\mathcal{B}$
(◇*n*[])	$\}\ n[\Diamond\mathcal{A}] \vdash \Diamond n[\mathcal{A}]$	(✧*n*[])	$\}\ n[$✧$\mathcal{A}] \vdash $✧$\mathcal{A}$
(◇ |)	$\}\ \Diamond\mathcal{A} \mid \Diamond\mathcal{B} \vdash \Diamond(\mathcal{A} \mid \mathcal{B})$	(✧ |)	$\}\ $✧$\mathcal{A} \mid \mathcal{B} \vdash $✧$(\mathcal{A} \mid \mathbf{T})$
(✧◇)	$\}\ $✧$\Diamond\mathcal{A} \vdash \Diamond$✧$\mathcal{A}$		

S4, but not S5: $\neg$ ***vld*** $\Diamond\mathcal{A} \vdash \Box\Diamond\mathcal{A}$ $\neg$ ***vld*** ✧$\mathcal{A} \vdash $¤✧$\mathcal{A}$

(✧◇): if somewhere sometime $\mathcal{A}$, then sometime somewhere $\mathcal{A}$

Equality

- Name equality can be defined within the logic:

$$\eta = \mu \;\triangleq\; \eta[\mathbf{T}]@\mu$$

Since (for any substitution applied to η,μ):

$P \vDash \eta[\mathbf{T}]@\mu$

iff $\mu[P] \vDash \eta[\mathbf{T}]$

iff $\eta = \mu \wedge P \vDash \mathbf{T}$

iff $\eta = \mu$

- Example: "Any two ambients here have different names":

$$\forall x.\forall y.\; x[\mathbf{T}] \mid y[\mathbf{T}] \mid \mathbf{T} \Rightarrow \neg\, x{=}y$$

Ex: Immovable Object vs. Irresistible Force

$$Im \triangleq \mathbf{T} \triangleright \Box(obj[\,] \mid \mathbf{T})$$

$$Ir \triangleq \mathbf{T} \triangleright \Box\Diamond\neg(obj[\,] \mid \mathbf{T})$$

$Im \mid Ir \vdash$	$(\mathbf{T} \triangleright \Box(obj[\,] \mid \mathbf{T})) \mid \mathbf{T}$	$\mathcal{A} \vdash \mathbf{T}$
$\vdash$	$\Box(obj[\,] \mid \mathbf{T})$	$(\mathcal{A} \triangleright \mathcal{B}) \mid \mathcal{A} \vdash \mathcal{B}$
$\vdash$	$\Diamond\Box(obj[\,] \mid \mathbf{T})$	$\mathcal{A} \vdash \Diamond\mathcal{A}$
$Im \mid Ir \vdash$	$\mathbf{T} \mid (\mathbf{T} \triangleright \Box\Diamond\neg(obj[\,] \mid \mathbf{T}))$	$\mathcal{A} \vdash \mathbf{T}$
$\vdash$	$\Box\Diamond\neg(obj[\,] \mid \mathbf{T})$	$\Diamond\neg\mathcal{A} \vdash \neg\Box\mathcal{A}$
$\vdash$	$\neg\Diamond\Box(obj[\,] \mid \mathbf{T})$	$\Box\neg\mathcal{A} \vdash \neg\Diamond\mathcal{A}$

Hence: $Im \mid Ir \vdash \mathbf{F}$ $\qquad$ $\mathcal{A} \wedge \neg\mathcal{A} \vdash \mathbf{F}$

Restriction

- $(\nu n)P$
  - "The name n is known only inside P."
  - "Create a new name n and use it in P."
  - It *extrudes* (floats) because it represents knowledge, not behavior:

$(\nu n)P \equiv (\nu m)(P\{n \leftarrow m\})$	a private name is as good as another
$(\nu n)\mathbf{0} \equiv \mathbf{0}$	
$(\nu n)(\nu m)P \equiv (\nu m)(\nu n)P$	
$(\nu n)(P \mid Q) \equiv (\nu n)P \mid Q$ if $n \notin fn(Q)$	
a.k.a. $(\nu n)(P \mid (\nu n)Q') \equiv (\nu n)P \mid (\nu n)Q'$	scope extrusion
$(\nu n)(m[P]) \equiv m[(\nu n)P]$ if $n \neq m$	

  - Used initially to represent private channels.
  - Later, to represent private names of any kind:
    Channels, Locations, Nonces, Cryptokeys, …

Revelation

$P \vDash n®\mathcal{A} \quad \triangleq \quad \exists P'\in\Pi.\ P \equiv (\nu n)P' \wedge P' \vDash \mathcal{A}$

- $n®\mathcal{A}$ is read, informally:
  - *Reveal* a private name as n and check that the revealed process satisfies $\mathcal{A}$.
  - Pull out (by extrusion) a (νn) binder, and check that the process stripped of the binder satisfies $\mathcal{A}$.
- Examples:
  - $n®n[\mathbf{0}]$: reveal a restricted name (say, p) as n and check the presence of an empty n location in the revealed process.

    $(\nu p)p[\mathbf{0}] \vDash n®n[\mathbf{0}]$

    because $(\nu p)p[\mathbf{0}] \equiv (\nu n)n[\mathbf{0}]$ and $n[\mathbf{0}] \vDash n[\mathbf{0}]$

Derived Formulas: Revelation

$\copyright n$	$\triangleq \neg n \circledR \mathbf{T}$	$P \vDash$ - iff $\neg\exists P' \in \Pi.\ P \equiv (\nu n)P'$
		iff $n \in fn(P)$
closed	$\triangleq \neg\exists x. \copyright x$	$P \vDash$ - iff $\neg\exists n \in \Lambda.\ n \in fn(P)$
separate	$\triangleq \neg\exists x. \copyright x \mid \copyright x$	$P \vDash$ - iff $\neg\exists n \in \Lambda, P' \in \Pi, P'' \in \Pi.$
		$P \equiv P' \mid P'' \wedge n \in fn(P') \wedge n \in fn(P'')$

- Examples:
  - $n[] \vDash \copyright n$
  - $(\nu p)p[] \vDash$ *closed*
  - $n[] \mid m[] \vDash$ *separate*

Revelation Rules

- Some mirror properties of restriction:

$$\} \;\; x \circledR x \circledR \mathcal{A} \dashv\vdash x \circledR \mathcal{A}$$

$$\} \;\; x \circledR y \circledR \mathcal{A} \dashv\vdash y \circledR x \circledR \mathcal{A}$$

$$\} \;\; x \circledR (\mathcal{A} \mid x \circledR \mathcal{B}) \dashv\vdash x \circledR \mathcal{A} \mid x \circledR \mathcal{B}$$ (scope extrusion)

- Some behave well with logical operators:

$$\} \;\; x \circledR (\mathcal{A} \vee \mathcal{B}) \vdash x \circledR \mathcal{A} \vee x \circledR \mathcal{A}$$

$$\mathcal{A} \vdash \mathcal{B} \;\; \} \;\; x \circledR \mathcal{A} \vdash x \circledR \mathcal{B}$$

- Some deal with the adjunction:

$$\eta \circledR \mathcal{A} \vdash \mathcal{B} \;\; \{\} \;\; \mathcal{A} \vdash \mathcal{B} \oslash \eta$$

$$\} \;\; (\neg \mathcal{A}) \oslash x \dashv\vdash \neg (\mathcal{A} \oslash x)$$

$$\} \;\; (\mathcal{A} \mid \mathcal{B}) \oslash x \vdash \mathcal{A} \oslash x \mid \mathcal{B} \oslash x$$

$$\} \;\; x \circledR ((\mathcal{A} \mid \mathcal{B}) \oslash x) \dashv\vdash x \circledR (\mathcal{A} \oslash x) \mid x \circledR (\mathcal{B} \oslash x)$$

Rules: Revelation

(®)	} x®x®$\mathcal{A} \dashv\vdash x$®$\mathcal{A}$	® idempotency
(® ®)	} x®y®$\mathcal{A} \vdash y$®x®$\mathcal{A}$	® commutativity
(® $\vee$)	} x®$(\mathcal{A} \vee \mathcal{B}) \vdash x$®$\mathcal{A} \vee x$®$\mathcal{A}$	®-$\vee$ distribution
(® $\vdash$)	$\mathcal{A} \vdash \mathcal{B}$ } x®$\mathcal{A} \vdash x$®$\mathcal{B}$	® congruence
(® $\obslash$)	η®$\mathcal{A} \vdash \mathcal{B}$ { } $\mathcal{A} \vdash \mathcal{B} \obslash \eta$	®-$\obslash$ adjunction
($\obslash$ $\neg$)	} $(\neg\mathcal{A}) \obslash x \dashv\vdash \neg(\mathcal{A} \obslash x)$	$\obslash$ is self-dual
($\obslash$ $\triangleright$**F**)	} $\mathcal{A}^{\mathbf{F}} \obslash x \dashv\vdash \mathcal{A}^{\mathbf{F}}$	$\obslash$ unsatisfiable

(® **0**)	}	x®**0** ⊣⊢ **0**		®/⦸-**0** rules
(⦸ **0**)	}	**0**⦸x ⊢ **0**		
(® |)	}	x®($\mathcal{A}$ | x®$\mathcal{B}$) ⊣⊢ x®$\mathcal{A}$ | x®$\mathcal{B}$		®/⦸-| rules
(⦸ |)	}	($\mathcal{A}$ | $\mathcal{B}$)⦸x ⊢ $\mathcal{A}$⦸x | $\mathcal{B}$⦸x		
(® ⦸ |)	}	x®(($\mathcal{A}$ | $\mathcal{B}$)⦸x) ⊢ x®($\mathcal{A}$⦸x) | x®($\mathcal{B}$⦸x)		
(® n[])	}	x®y[$\mathcal{A}$] ⊣⊢ y[x®$\mathcal{A}$]	($x \neq y$)	®/⦸-n[-] rules
(⦸ n[])	}	y[$\mathcal{A}$]⦸x ⊢ y[$\mathcal{A}$⦸x]	($x \neq y$)	
(⦸ n[])	}	x[$\mathcal{A}$]⦸x ⊢ **F**		

Fresh-Name Quantifier

$P \vDash \text{И}x.\mathcal{A} \quad \triangleq \quad \exists m\in\Lambda.\ m\notin fn(P,\mathcal{A}) \wedge P \vDash \mathcal{A}\{x\leftarrow m\}$

- *C.f.*: $P \vDash \exists x.\mathcal{A}$ iff $\exists m\in\Lambda.\ P \vDash \mathcal{A}\{x\leftarrow m\}$
- Actually definable (metatheoretically, as an abbreviation):

  $\text{И}x.\mathcal{A} \ \triangleq\ \exists x.\ x\#(fnv(\mathcal{A})\text{-}\{x\}) \wedge x®\mathbf{T} \wedge \mathcal{A}$

  Provided we add the axiom schema:

  (GP) $\quad \vdash\ \exists x.\ x\#N \wedge x®\mathbf{T} \wedge \mathcal{A} \dashv\vdash \forall x.\ (x\#N \wedge x®\mathbf{T}) \Rightarrow \mathcal{A}$

  where $N \supseteq fnv(\mathcal{A})\text{-}\{x\}$ and $x\notin N$

- Fundamental "freshness" property (Gabbay-Pitts):

  $\text{И}x.\mathcal{A}$ iff $\exists m\in\Lambda.\ m\notin fn(P,\mathcal{A}) \wedge P \vDash \mathcal{A}\{x\leftarrow m\}$

  iff $\forall m\in\Lambda.\ m\notin fn(P,\mathcal{A}) \Rightarrow P \vDash \mathcal{A}\{x\leftarrow m\}$

because *any fresh name is as good as any other*.

- Very nice logical properties:
  - $\forall x.\mathcal{A} \vdash И x.\mathcal{A} \vdash \exists x.\mathcal{A}$

    $\neg И x.\mathcal{A} \dashv\vdash И x.\neg\mathcal{A}$
  - $И x.(\mathcal{A} \mid \mathcal{B}) \dashv\vdash (И x.\mathcal{A}) \mid (И x.\mathcal{B})$ (hint: (GP) $\exists$ for $\Rightarrow$, $\forall$ for $\Leftarrow$)
  - $\Diamond И x.\mathcal{A} \dashv\vdash И x.\Diamond\mathcal{A}$

Hidden-Name Quantifier

$\mathrm{H}x.\mathcal{A} \quad \triangleq \quad \text{И}x.x®\mathcal{A}$

$P \vDash \mathrm{H}x.\mathcal{A}$ iff

$\exists m \in \Lambda, P' \in \Pi.\ m \notin fn(\mathcal{A}) \wedge P \equiv (\nu m)P' \wedge P' \vDash \mathcal{A}\{x \leftarrow m\}$

- Example: $\mathrm{H}x.x[\,] = \text{И}x.x®x[\,]$
  - "for hidden x, we find a void location called x" = "for fresh x, we reveal a hidden name as x, then we find a void location x"
  - $(\nu n)n[\,] \vDash \mathrm{H}x.x[\,]$ because $(\nu n)n[\,] \vDash \text{И}x.x®x[\,]$ because $(\nu n)n[\,] \vDash n®n[\,]$ (where $n \notin fn((\nu n)n[\,])$).
- Counterexamples:
  - $(\nu m)m[\,] \nvDash \mathrm{H}x.n[\,]$ (N.B.: this holds for $\mathrm{H}x.\mathcal{A} \triangleq \exists x.x®\mathcal{A}$!)
  - $(\nu n)n[\,] \mid (\nu n)n[\,] \nvDash \mathrm{H}x.(x[\,] \mid x[\,])$
  - $(\nu n)(n[\,] \mid n[\,]) \nvDash \mathrm{H}x.x[\,] \mid \mathrm{H}x.x[\,]$

Forget $n \circledR \mathcal{A}$ and $\text{И}x.\mathcal{A}$, why not just use $\mathrm{H}x.\mathcal{A}$?

- Consider:

  $\text{И}x.x \circledR (\mathcal{A} \mid x \circledR \mathcal{B})$

  $\dashv\vdash \text{И}x.(x \circledR \mathcal{A} \mid x \circledR \mathcal{B})$

  $\dashv\vdash (\text{И}x.x \circledR \mathcal{A}) \mid (\text{И}x.x \circledR \mathcal{B})$

- That is:

  $\mathrm{H}x.(\mathcal{A} \mid x \circledR \mathcal{B}) \dashv\vdash \mathrm{H}x.\mathcal{A} \mid \mathrm{H}x.\mathcal{B}$

- Hence, the scope extrusion rule for H still uses $\circledR$.

- No matter what one choses as primitives, we have explored interesting connections between these operators. ($\text{И}+\circledR$ and $\mathrm{H}+\copyright$ are *almost* interdefinable [Caires].)

Example: Key Sharing

- Consider a situation where "a hidden name x is shared by two locations n and m, and is <u>not known</u> outside those locations".

$$\text{H}x.(n[\copyright x] \mid m[\copyright x])$$

  - $P \vDash \text{H}x.(n[\copyright x] \mid m[\copyright x])$

    $\Leftrightarrow \exists r \in \Lambda.\ r \notin fn(P) \cup \{n,m\} \wedge \exists R', R'' \in \Pi.\ P \equiv (\nu r)(n[R'] \mid m[R''])$
    $\wedge\ r \in fn(R') \wedge r \in fn(R'')$
  - E.g.: take $P = (\nu p)\ (n[p[]] \mid m[p[]])$.

- A protocol establishing a shared key should satisfy:

$$\Diamond \text{H}x.(n[\copyright x] \mid m[\copyright x])$$

From Logic back to Types

- A logic is *just a very rich type system*.
    - Type systems are very "structural" (i.e., the structure of types reflects closely the structure of values). Our logic is extremely structural (intensional) for a logic. It is in fact almost as structural as a type system.
    - Type systems for process calculi often have a parallel composition operation on types. I.e., they are "spatial" in our sense.
    - Therefore, our work may help in discerning patterns in the large and diverse collection of type systems for process calculi. These usually become particularly tangled when trying to handle restriction.
- Suppose that $P : \mathcal{A}$ means that process P may have "effects" $\mathcal{A}$, where an effect is any kind of information about the behavior of P that one may want to track statically. Then the following kind of typing rules happen:
    - Effects may be composed:
      $\Gamma \vdash P : \mathcal{A}\,,\ \Gamma \vdash Q : \mathcal{B}\ \}\ \Gamma \vdash P \mid Q : \mathcal{A} \mid \mathcal{B}$
    - Effects may be hidden:
      $\Gamma, n{:}\mathcal{A} \vdash P : \mathcal{B}\{x{\leftarrow}n\}\ \}\ \Gamma \vdash (\nu n{:}\mathcal{A})P : \text{H}x{:}\mathcal{A}.\mathcal{B}$
    - *C.f.* Kobayashi: behavioral type systems. *C.f.* "exchange types" for Ambients.

Applications

- Modelchecking security+mobility assertions:
  - If P is $!$-free and $\mathcal{A}$ is $\triangleright$-free, then $P \vDash \mathcal{A}$ is decidable.
  - This provides a way of mechanically checking (certain) assertions about (certain) mobile processes.
  - Expressing mobility/security policies of host sites. (Conferring more flexibility than just sandboxing the agent.)
  - Just-in-time verification of code containing mobility instructions (by either modelchecking or proof-carrying code).
- Expressing properties of type systems (beyond subject reduction).
  - Expressing Locking
    - If $E, n{:}Amb^{\bullet}[S] \vdash P : T$ (a typing judgment asserting that no ambient called n can ever be opened in P), then:

      $$P \vDash \Box(\Diamond an\ n \Rightarrow \Box\Diamond an\ n)$$
  - Expressing Immobility
    - If $E, p{:}Amb^{\bullet}[S], q{:}Amb^{\bullet}[{}^{\curlyvee}S'] \vdash P : T$ (a typing judgment asserting that no ambient called q can ever move within P), then:

      $$P \vDash \Box(\Diamond(p\ parents\ q) \Rightarrow \Box\Diamond(p\ parents\ q))$$

      where $p\ parents\ q \ \triangleq\ p[q[\mathbf{T}] \mid \mathbf{T}] \mid \mathbf{T}$

Conclusions

- The novel aspects of our logic lie in its explicit treatment of space and of the evolution of space over time (mobility).
- We can now talk also about fresh and hidden locations.
- These ideas can be applied to any process calculus that embodies a distinction between spatial and temporal operators, and a restriction operator.
- Our logical rules arise from a particular model. This approach makes the logic very concrete (and sound), but raises questions of logical completeness.

Exercise

- Show that $\rbrace\ (\mathcal{A} \mid \mathcal{B}) \wedge \mathbf{0} \vdash \mathcal{A}$ is valid (by applying the definition of sequent and of satisfaction). The proof is short. In the process, you will discover you need a little ambient calculus lemma about $P \mid Q \equiv \mathbf{0}$; you do not need to prove it but you need to identify it.
- (Hard/Optional)
  Find a formal derivation of $\rbrace\ (\mathcal{A} \mid \mathcal{B}) \wedge \mathbf{0} \vdash \mathcal{A}$ from the axioms in the slides.
  (My) proof uses the decomposition axiom, $(\mid \|)$.

END

Semantics

- Version 1 [Cardelli-Gordon]
  - For the restriction-free ambient calculus.
  - Formulas denote sets of processes that are closed under structural congruence.
- Version 2 [Cardelli-Gordon]
  - For the ambient calculus with restriction.
  - Formulas denote sets of processes that are closed under structural congruence. Freshness handled “metatheoretically”.
- Version 3 [Caires-Cardelli]
  - To handle both restriction and recursive formulas, and to handle freshness “properly”. (For the π-calculus, for simplicity.)
  - Formulas denote sets of processes that are closed under congruence and that have *finite support* (are closed under transpositions outside of a finite set *N* of names).

A Good Property

- A property not shared by other candidate definitions, such as $\exists x.x®\mathcal{A}$ and $\forall x.x®\mathcal{A}$. This is even derivable within the logic:

$$\text{И}x.(\mathcal{A}\{n\leftarrow x\}) \wedge n®\mathbf{T} \dashv\vdash n®\mathcal{A} \qquad \text{where } x \notin fv(\mathcal{A})$$

- It implies:

$$P \vDash \mathcal{A} \;\Rightarrow\; (\nu n)P \vDash \text{И}x.(\mathcal{A}\{n\leftarrow x\})$$

$$P \vDash \text{И}x.(\mathcal{A}\{n\leftarrow x\}) \wedge n \notin fn(P) \;\Rightarrow\; P \vDash n®\mathcal{A}$$

$$P \vDash n®\mathcal{A} \;\Rightarrow\; P \vDash \text{И}x.(\mathcal{A}\{n\leftarrow x\})$$

A Surprising Property

$\mathsf{H}x.\mathcal{A} \nvdash \mathcal{A}$ for $x \notin fv(\mathcal{A})$

- Ex.: $\mathsf{H}x.(\neg\mathbf{0} \mid \neg\mathbf{0}) \nvdash \neg\mathbf{0} \mid \neg\mathbf{0}$

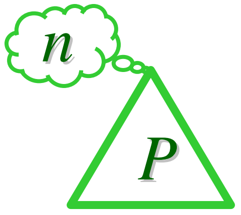
  If for a hidden x the inner system can be decomposed into two non-void parts, it does not mean that the whole system can be decomposed, because the two parts may be entangled by restriction:

  $(\nu n)(n[] \mid n[]) \vDash \text{И}x.x®(\neg\mathbf{0} \mid \neg\mathbf{0})$ but:

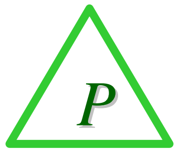
  $(\nu n)(n[] \mid n[]) \nvDash \neg\mathbf{0} \mid \neg\mathbf{0}$.
- This is ®'s fault, not И's: with the same counterexample we can show $n®(\neg\mathbf{0} \mid \neg\mathbf{0}) \nvdash \neg\mathbf{0} \mid \neg\mathbf{0}$.
- However, $\mathsf{H}x.\mathbf{0} \vdash \mathbf{0}$.
- Moreover, $\mathcal{A} \vdash \mathsf{H}x.\mathcal{A}$ for $x \notin fv(\mathcal{A})$.

Satisfaction for Hidden-Name Quantification

- It makes sense also to define a *hidden name quantifier* $\mathrm{H}x.\mathcal{A}$:
  - n®$\mathcal{A}$: reveal a hidden name <u>if possible</u> as a given n, and assert $\mathcal{A}\{n\}$.
  - $\mathrm{H}x.\mathcal{A}$: reveal a hidden name as <u>any fresh</u> name x and assert $\mathcal{A}\{x\}$.



$\vDash \mathrm{H}x.\mathcal{A}$ if



$\vDash \mathcal{A}\{x\leftarrow n\}$

with $n \notin fn(\mathcal{A})$

- Design decision: how to define $\mathrm{H}x.\mathcal{A}$, keeping in mind that "freshness" may spill into the logic?
  - *The Obvious Thing*: extend the syntax with $\mathrm{H}x.\mathcal{A}$ and define it directly.
  - *Luis Caires:* Extend the syntax with $\mathrm{H}x.\mathcal{A}$ and add signatures to keep track of free names, to enforce the side condition $n \notin fn(\mathcal{A})$: $\Sigma \bullet P \vDash \Sigma \bullet \mathcal{A}$.
  - *Us:* Retain n®$\mathcal{A}$ and mix it with a logical notions of freshness $\text{И}x.\mathcal{A}$ (one extra axiom schema, no new syntax). We eventually define: $\mathrm{H}x.\mathcal{A} \triangleq \text{И}x.x$®$\mathcal{A}$.

The Decomposition Operator

- Consider the De Morgan dual of | :

$\mathcal{A} \parallel \mathcal{B}$	$\triangleq \neg(\neg\mathcal{A} \mid \neg\mathcal{B})$	$P \vDash$ - iff $\forall P', P'' \in \Pi.\ P \equiv P' \mid P'' \Rightarrow P' \vDash \mathcal{A} \vee P'' \vDash \mathcal{B}$
$\mathcal{A}^{\forall}$	$\triangleq \mathcal{A} \parallel \mathbf{F}$	$P \vDash$ - iff $\forall P', P'' \in \Pi.\ P \equiv P' \mid P'' \Rightarrow P' \vDash \mathcal{A}$
$\mathcal{A}^{\exists}$	$\triangleq \mathcal{A} \mid \mathbf{T}$	$P \vDash$ - iff $\exists P', P'' \in \Pi.\ P \equiv P' \mid P'' \wedge P' \vDash \mathcal{A}$

$\mathcal{A} \parallel \mathcal{B}$	for every partition, one piece satisfies $\mathcal{A}$ or the other piece satisfies $\mathcal{B}$
$\mathcal{A}^{\forall} \Leftrightarrow \neg((\neg\mathcal{A})^{\exists})$	every component satisfies $\mathcal{A}$
$\mathcal{A}^{\exists} \Leftrightarrow \neg((\neg\mathcal{A})^{\forall})$	some component satisfies $\mathcal{A}$

Examples:

$(p[\mathbf{T}] \Rightarrow p[q[\mathbf{T}]^{\exists}])^{\forall}$	every p has a q child
$(p[\mathbf{T}] \Rightarrow p[q[\mathbf{T}] \mid (\neg q[\mathbf{T}])^{\forall}])^{\forall}$	every p has a unique q child

The Decomposition Axiom

(| ||) } $(\mathcal{A}' \mid \mathcal{A}'') \vdash (\mathcal{A}' \mid \mathcal{B}'') \vee (\mathcal{B}' \mid \mathcal{A}'') \vee (\neg\mathcal{B}' \mid \neg\mathcal{B}'')$

- Alternative formulations and special cases:

} $(\mathcal{A}' \mid \mathcal{A}'') \wedge (\mathcal{B}' \parallel \mathcal{B}'') \vdash (\mathcal{A}' \mid \mathcal{B}'') \vee (\mathcal{B}' \mid \mathcal{A}'')$

"If ***P*** has a partition into pieces that satisfy $\mathcal{A}'$ and $\mathcal{A}''$, and every partition has one piece that satisfies $\mathcal{B}'$ or the other that satisfies $\mathcal{B}''$, then either ***P*** has a partition into pieces that satisfy $\mathcal{A}'$ and $\mathcal{B}''$, or it has a partition into pieces that satisfy $\mathcal{B}'$ and $\mathcal{A}''$."

} $\neg(\mathcal{A} \mid \mathcal{B}) \vdash (\mathcal{A} \mid \mathbf{T}) \Rightarrow (\mathbf{T} \mid \neg\mathcal{B})$

"If ***P*** has no partition into pieces that satisfy $\mathcal{A}$ and $\mathcal{B}$, but ***P*** has a piece that satisfies $\mathcal{A}$, then ***P*** has a piece that does not satisfy $\mathcal{B}$."

} $\neg(\mathbf{T} \mid \mathcal{B}) \vdash \mathbf{T} \mid \neg\mathcal{B}$

} $\neg(\mathcal{A} \mid \mathcal{B}) \vdash (\neg\mathcal{A} \mid \mathbf{T}) \vee (\mathbf{T} \mid \neg\mathcal{B})$

Logical Adjunctions

- This is a logic with multiple logical adjunctions (4 of them!):

  - $\wedge$ / $\Rightarrow$ (classical)

    $\mathcal{A} \wedge \mathcal{C} \vdash \mathcal{B}$ iff $\mathcal{A} \vdash \mathcal{C} \Rightarrow \mathcal{B}$

  - $\mid$ / $\triangleright$ (linear, $\otimes$ / $\multimap$)

    $\mathcal{A} \mid \mathcal{C} \vdash \mathcal{B}$ iff $\mathcal{A} \vdash \mathcal{C} \triangleright \mathcal{B}$

  - $n[\text{-}]$ / $\text{-}@n$

    $n[\mathcal{A}] \vdash \mathcal{B}$ iff $\mathcal{A} \vdash \mathcal{B}@n$

  - n®- / $\text{-}\oslash n$

    n®$\mathcal{A} \vdash \mathcal{B}$ iff $\mathcal{A} \vdash \mathcal{B}\oslash n$

- Which one should be taken as *the* logical adjunction for sequents? (I.e., what should "," mean in a sequent?)
- We do not choose, and take sequents of the form $\mathcal{A} \vdash \mathcal{B}$.

“Neutral” Sequents

- Our logic is formulated with single-premise, single-conclusion sequents. We don’t pre-judge “,”.
  - By taking $\wedge$ on the left and $\vee$ on the right of $\vdash$ as structural operators, we can derive all the standard rules of sequent and natural deduction systems with multiple premises/conclusions.
  - By taking $\mid$ on the left of $\vdash$ as a structural operator, we can derive all the rules of intuitionistic linear logic (by appropriate mappings of the ILL connectives).
  - By taking nestings of $\wedge$ and $\mid$ on the left of $\vdash$ as structural “bunches”, we obtain a bunched logic, with its two associated implications, $\Rightarrow$ and $\triangleright$.
- This is convenient. We do not know much, however, about the meta-theory of this presentation style.

Ambient Calculus: Example

$$\overbrace{a[msg[\langle M\rangle \mid out\ a.\ in\ b.\ P]]}^{location\ a}\ \mid \overbrace{b[open\ msg.\ (n).\ P]}^{location\ b}$$

$$\underbrace{a[msg[\langle M\rangle \mid out\ a.\ in\ b.\ P]]}_{send\ M\ from\ a\ to\ b}\ \mid b[\underbrace{open\ msg.\ (n).\ P}_{receive\ n;\ do\ P}]$$

The packet *msg* moves from *a* to *b*, mediated by the capabilities *out a* (to exit *a*), *in b* (to enter *b*), and *open msg* (to open the *msg* envelope).

$$
\begin{array}{lll}
 & a[msg[\langle M\rangle \mid out\ a.\ in\ b.\ P]] & \mid b[open\ msg.\ (n).\ P] \\
\text{(exit)} \longrightarrow & a[] \quad \mid msg[\langle M\rangle \mid in\ b.\ P] & \mid b[open\ msg.\ (n).\ P] \\
\text{(enter)} \longrightarrow & a[] & \mid b[msg[\langle M\rangle] \mid open\ msg.\ (n).\ P] \\
\text{(open)} \longrightarrow & a[] & \mid b[\langle M\rangle \mid (n).\ P] \\
\text{(read)} \longrightarrow & a[] & \mid b[P\{n\leftarrow M\}]
\end{array}
$$

Connections with Intuitionistic Linear Logic

- Weakening and contraction are not valid rules: principle of *conservation of space*.
- Semantic connection: sets of processes closed under $\equiv$ and ordered by inclusion form a quantale (a model of ILL).
- Multiplicative intuitionistic linear logic (MILL) can be faithfully embedded in our logic:

$$\begin{aligned}
\mathbf{1}_{\mathrm{MILL}} &\triangleq \mathbf{0} \\
\mathcal{A} \otimes_{\mathrm{MILL}} \mathcal{B} &\triangleq \mathcal{A} \mid \mathcal{B} \\
\mathcal{A} \multimap_{\mathrm{MILL}} \mathcal{B} &\triangleq \mathcal{A} \triangleright \mathcal{B}
\end{aligned}$$

MILL rules and our rules are interderivable ("our rules" means the rules involving only $\mathbf{0}$, $\mid$, $\triangleright$, plus a derivable cut rule for $\mid$).

- Full intuitionistic linear logic (ILL) can be embedded:

$$
\begin{array}{lll@{\qquad\qquad}lll}
\mathbf{1}_{\mathrm{ILL}} & \triangleq & \mathbf{0} & \mathcal{A} \oplus \mathcal{B} & \triangleq & \mathcal{A} \vee \mathcal{B} \\
\bot_{\mathrm{ILL}} & \triangleq & \mathbf{F} & \mathcal{A} \,\&\, \mathcal{B} & \triangleq & \mathcal{A} \wedge \mathcal{B} \\
\top_{\mathrm{ILL}} & \triangleq & \mathbf{T} & \mathcal{A} \otimes \mathcal{B} & \triangleq & \mathcal{A} \mid \mathcal{B} \\
\mathbf{0}_{\mathrm{ILL}} & \triangleq & \mathbf{F} & \mathcal{A} \multimap \mathcal{B} & \triangleq & \mathcal{A} \triangleright \mathcal{B} \\
 & & & !\mathcal{A} & \triangleq & \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}
\end{array}
$$

- The rules of ILL can be logically derived from these definitions. (E.g.: the proof of $!\mathcal{A} \vdash !\mathcal{A} \otimes !\mathcal{A}$ uses the decomposition axiom.)

- So, $\mathcal{A}_1, \ldots, \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B}$ implies $\mathcal{A}_1 \mid \ldots \mid \mathcal{A}_n \vdash \mathcal{B}$.

- Some discrepancies: $\bot_{\mathrm{ILL}} = \mathbf{0}_{\mathrm{ILL}}$; the additives distribute; $!\mathcal{A}$ is not "replication"; $!\mathcal{A} \multimap \mathcal{B}$ is not so interesting; $\mathcal{A}^{\bot}/\mathcal{A}^{\mathbf{0}}$ is unusually interesting.

Connection with Relevant Logic

- (Noted after the fact [O'Hearn, Pym].) The definition of the satisfaction relation is very similar to Urquhart's semantics of relevant logic. In particular $\mathcal{A} \mid \mathcal{B}$ is defined just like *intensional conjunction*, and $\mathcal{A} \triangleright \mathcal{B}$ is defined just like *relevant implication* in that semantics.

- Except:
  - We do not have contraction. This does not make sense in process calculi, because $P \mid P \neq P$. Urquhart semantics without contraction does not seem to have been studied.
  - We use an equivalence $\equiv$, instead of a Kripke-style partial order $\emptyset$ as in Urquhart's general case. (We may have a need for a partial order in more sophisticated versions of our logic.)

Connections with Bunched Logic

- Peter O'Hearn and David Pym study *bunched logics*, where sequents have two structural combinators, instead of the standard single "," combinator (usually meaning $\wedge$ or $\otimes$ on the left) found in most presentations of logic. Thus, sequents are *bunches* of formulas, instead of lists of formulas. Correspondingly, there are two implications that arise as the adjuncts of the two structural combinators.

- The situation is very similar to our combinators $\mid$ and $\wedge$, which can combine to irreducible bunches of formulas in sequents, and to our two implications $\Rightarrow$ and $\rhd$. However, we have a classical and a linear implication, while bunched logics have so far had an intuitionistic and a linear implication.

Semantic Connections with the Linear Logic

- A (commutative) quantale $\mathcal{Q}$ is a structure

  $<S \in \mathrm{Set}, \leq \in S^2 \to \mathrm{Bool}, \bigvee \in \mathcal{P}(S) \to S, \otimes \in S^2 \to S, 1 \in S>$ such that:

  $\leq, \bigvee$ is a complete join semilattice

  $\otimes, 1$ is a commutative monoid

  $p \otimes \bigvee Q = \bigvee \{p \otimes q \mid q \in Q\}$

- They are complete models of Intuitionistic Linear Logic (ILL):

$[\![\mathcal{A} \oplus \mathcal{B}]\!]$	$\triangleq$	$\bigvee \{[\![\mathcal{A}]\!], [\![\mathcal{B}]\!]\}$	$[\![\mathbf{1}_{\mathrm{ILL}}]\!]$	$\triangleq$	1
$[\![\mathcal{A} \& \mathcal{B}]\!]$	$\triangleq$	$\bigvee \{C \mid C \leq [\![\mathcal{A}]\!] \wedge C \leq [\![\mathcal{B}]\!]\}$	$[\![\bot_{\mathrm{ILL}}]\!]$	$\triangleq$	any element of S
$[\![\mathcal{A} \otimes \mathcal{B}]\!]$	$\triangleq$	$[\![\mathcal{A}]\!] \otimes [\![\mathcal{B}]\!]$	$[\![\top_{\mathrm{ILL}}]\!]$	$\triangleq$	$\bigvee S$
$[\![\mathcal{A} \multimap \mathcal{B}]\!]$	$\triangleq$	$\bigvee \{C \mid C \otimes [\![\mathcal{A}]\!] \leq [\![\mathcal{B}]\!]\}$	$[\![\mathbf{0}_{\mathrm{ILL}}]\!]$	$\triangleq$	$\bigvee \emptyset$

$[\![!\mathcal{A}]\!] \triangleq \upsilon X.\, [\![\mathbf{1} \& \mathcal{A} \& X \otimes X]\!]$ where $\upsilon X.\, A\{X\} \triangleq \bigvee \{C \mid C \leq A\{C\}\}$

$$\mathbf{vld}_{\mathrm{ILL}}(\mathcal{A}_1, \ldots, \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B})_{\mathcal{Q}} \triangleq [\![\mathcal{A}_1]\!]_{\mathcal{Q}} \otimes_{\mathcal{Q}} \ldots \otimes_{\mathcal{Q}} [\![\mathcal{A}_n]\!]_{\mathcal{Q}} \leq_{\mathcal{Q}} [\![\mathcal{B}]\!]_{\mathcal{Q}}$$

The Process Quantale

- The sets of processes closed under $\equiv$ and ordered by inclusion form a quantale (let $A^{\equiv} \triangleq \{P \mid\mid \exists Q{\in}A.\ P{\equiv}Q\}$):

  $\Theta \triangleq \langle\Phi, \subseteq, \bigcup, \otimes, \mathbf{1}\rangle$ where, for $A,B \subseteq \Pi$:

  $\Phi \triangleq \{A^{\equiv} \mid\mid A \subseteq \Pi\}$

  $1_{\Theta} \triangleq \{\mathbf{0}\}^{\equiv}$

  $A \otimes_{\Theta} B \triangleq \{P \mid Q \mid\mid P \in A \wedge Q \in B\}^{\equiv}$

- ILL validity in Θ:

  $\mathbf{vld}_{\mathrm{ILL}}(\mathcal{A}_1, \ldots, \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B})_{\Theta}$

  $\Leftrightarrow\ [\![\mathcal{A}_1]\!] \otimes_{\Theta} \ldots \otimes_{\Theta} [\![\mathcal{A}_n]\!] \subseteq [\![\mathcal{B}]\!]$

  $\Leftrightarrow\ [\![\mathcal{A}_1 \mid \ldots \mid \mathcal{A}_n]\!] \subseteq [\![\mathcal{B}]\!]$

  $\Leftrightarrow\ (\Pi - [\![\mathcal{A}_1 \mid \ldots \mid \mathcal{A}_n]\!]) \cup [\![\mathcal{B}]\!] = \Pi$

  $\Leftrightarrow\ [\![\mathcal{A}_1 \mid \ldots \mid \mathcal{A}_n \Rightarrow \mathcal{B}]\!] = \Pi$

Process Domain

- Semantic domain: Θ

$$
\begin{array}{lll}
 & \Pi & \triangleq \text{ the set of process expressions} \\
\forall C \subseteq \Pi. & C^{\equiv} & \triangleq \{P \in \Pi \mid \exists P' \in C.\ P' \equiv P\} \\
 & \Phi & \triangleq \{C^{\equiv} \mid C \subseteq \Pi\}
\end{array}
$$

The domain Θ is both a quantale $(1, \otimes, \subseteq, \bigcup)$ and a boolean algebra $(\emptyset, \Pi, \cup, \cap, \Pi^{-})$. It has additional structure induced by $n[P]$ and $(\nu n)P$.

- Spatial operators over Θ:

$$
\begin{array}{lll}
 & 1 & \triangleq \{\mathbf{0}\}^{\equiv} \\
\forall C,D \in \Theta. & C \otimes D & \triangleq \{P|Q \mid P \in C \wedge Q \in D\}^{\equiv} \\
\forall n \in \Lambda,\ C \in \Theta. & n[C] & \triangleq \{n[P] \mid P \in C\}^{\equiv} \\
\forall n \in \Lambda,\ C \in \Theta. & n ® C & \triangleq \{(\nu n)P \mid P \in C\}^{\equiv}
\end{array}
$$

Semantics of Revelation

$n®C \ \triangleq \ \{(\nu n)P \mid P \in C\}^{\equiv}$

- This means: take all processes of the form $(\nu n)P$ (*not* up to renaming of n), remove the ones such that $P \notin C$, and $\equiv$-close the result (thus adding all the α-variants).
- $n®C$ is read, informally:
  - *Reveal* a private name as n and check that the contents are in C.
  - Pull (by $\equiv$) a (νn) binder at the top and check the rest is in C.
- Ex.: $n®n[1]$: reveal a private name (say, p) as n and check that there is an empty n ambient in the revealed process.

  $(\nu p)p[\mathbf{0}] \in n®n[1]$

  because $(\nu p)p[\mathbf{0}] \equiv (\nu n)n[\mathbf{0}]$ and $n[\mathbf{0}] \in n[1]$

- More examples of $n®C \triangleq \{(\nu n)P \mid P \in C\}^{\equiv}$:
  - $\mathbf{0} \in n®1$ because $\mathbf{0} \equiv (\nu n)\mathbf{0}$ and $\mathbf{0} \in 1$
  - $m[\mathbf{0}] \in n®\Pi$ because $m[\mathbf{0}] \equiv (\nu n)m[\mathbf{0}]$ and $m[\mathbf{0}] \in \Pi$
  - $n[\mathbf{0}] \notin n®\Pi$ because $n[\mathbf{0}] \not\equiv (\nu n)$...

- Therefore, $n®C$ is:
  - closed under α-variants
  - closed under $\equiv$-variants
  - not closed under changes in the set of free names
  - not closed under reduction (free names may disappear)
  - not closed under any equivalence that includes reduction
  - still ok for temporal reasoning: $\neg n®\mathcal{A} \wedge \Diamond n®\mathcal{A}$

Semantics of the Logic

$$
\begin{array}{lcl}
[\![\mathbf{T}]\!] & \triangleq & \Pi \\
[\![\neg\mathcal{A}]\!] & \triangleq & \Pi - [\![\mathcal{A}]\!] \\
[\![\mathcal{A} \vee \mathcal{B}]\!] & \triangleq & [\![\mathcal{A}]\!] \cup [\![\mathcal{B}]\!] \\
[\![\mathbf{0}]\!] & \triangleq & \mathbf{1} \\
[\![n[\mathcal{A}]]\!] & \triangleq & n[[\![\mathcal{A}]\!]] \\
[\![\mathcal{A}@n]\!] & \triangleq & \bigcup\{C \in \Theta \mid n[C] \subseteq [\![\mathcal{A}]\!]\} \\
[\![\mathcal{A} \mid \mathcal{B}]\!] & \triangleq & [\![\mathcal{A}]\!] \otimes [\![\mathcal{B}]\!] \\
[\![\mathcal{A} \triangleright \mathcal{B}]\!] & \triangleq & \bigcup\{C \in \Theta \mid C \otimes [\![\mathcal{A}]\!] \subseteq [\![\mathcal{B}]\!]\} \\
[\![n \circledR \mathcal{A}]\!] & \triangleq & n \circledR [\![\mathcal{A}]\!] \\
[\![\mathcal{A} \oslash n]\!] & \triangleq & \bigcup\{C \in \Theta \mid n \circledR C \subseteq [\![\mathcal{A}]\!]\} \\
[\![\text{✧}\mathcal{A}]\!] & \triangleq & \{P \in \Pi \mid \exists P' \in \Pi.\ P \downarrow^{*} P' \wedge P' \in [\![\mathcal{A}]\!]\} \\
[\![\Diamond\mathcal{A}]\!] & \triangleq & \{P \in \Pi \mid \exists P' \in \Pi.\ P \longrightarrow^{*} P' \wedge P' \in [\![\mathcal{A}]\!]\} \\
[\![\forall x.\mathcal{A}]\!] & \triangleq & \bigcap_{m \in \Lambda} [\![\mathcal{A}\{x \leftarrow m\}]\!]
\end{array}
$$

$P \downarrow P'$ iff $\exists n, P''.\ P \equiv n[P'] \mid P''$; $\downarrow^{*}$ is the refl-trans closure of $\downarrow$

Basic Fact

- Formulas describe only congruence-invariant properties:

$$\forall \mathcal{A} \in \Phi.\ [\![\mathcal{A}]\!] \in \Theta$$

Recovering the Satisfaction Relation

$$P \vDash \mathcal{A} \quad \triangleq \quad P \in [\![\mathcal{A}]\!]$$

- The properties of satisfaction for each logic constructs are then derivable.
- This approach to defining satisfaction is particularly good for introducing recursive formulas in the logic: it is easy to give them semantics as least and greatest fixpoints in the model, while it is not easy to define them directly via a satisfaction relation.