Spatial Logics for Distributed Systems

Luca Cardelli

Microsoft Research

Oxford, Strachey Lecture, 2002-04-23

Reflecting joint work with
Luís Caires, Philippa Gardner, Andrew D. Gordon, Giorgio Ghelli.

Widely Distributed Systems

Concurrent systems that are *spatially* distributed

- Not in the same box.
- Not on the same LAN.
- Not inside the same firewall.
- Not always in the same place.

They have well-defined subsystems that:

- Fail independently.
- Recover independently.
- Hold secrets, mistrust each other.
- Move around.

The New Machine

The “machine” we now write programs for, is the whole Internet.

- New instruction sets (programming models):
    - Message-centric, asynchronous, often stateless.
      Cannot rely on distributed consensus.
    - In striking contrast to shared-memory concurrency, and handshake-based (synchronous) concurrency.
- New type systems:
    - Traditional “strong” type systems have been (finally!) enthusiastically adopted as a foundation for security.
    - But entirely new type systems are needed for regulating communication, and to manage application-level security.
- New program logics:
    - Privacy/security concerns override everything else.
    - Need “location awareness”.

Talking About *Where*

Informal statements:

- Distribution: *Where* are things happening?
- Security: *Where* are things kept, and who can get there?
- Privacy: *Where* are things known, and where are they leaked?

We need a new way of reasoning (i.e. a new logic):

- Classical logic: *Whether* something is true.
- Intuitionistic logic: *How* something is true.
- Temporal logic: *When* something is true.
- *Spatial* logic: *Where* something is true.

Outline

We look, concretely, at specific logics for specific models:

- For trees.
- For graphs.
- For mobility.
- For communication.
- For privacy.

With some common, new-ish, techniques:

- Semantically: Modal logics for structured worlds.
- Syntactically: Many-world sequent calculi.

1: A Logic for Trees

Historically, this all began with a spatial logics for the Ambient Calculus (a process calculus based on trees of locations).

As a tutorial, we start by looking just at the trees.

Spatial interpretation: a formula holds at a particular (sub-)tree.

This has also its own applications.

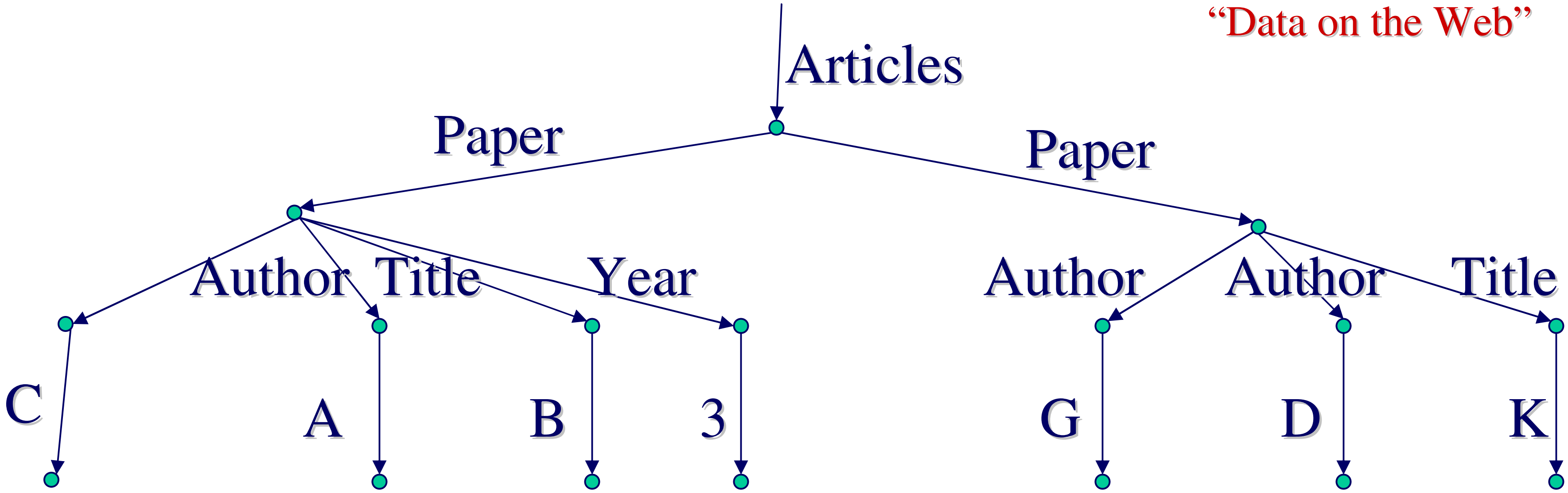
Cardelli: ***Describing Semistructured Data.*** SIGMOD Record.

Cardelli-Ghelli: ***A Query Language based on the Ambient Logic***. ESOP'01.

Semistructured Data

(I.e.: XML after parsing)

Abiteboul, Buneman, Suciu:
"Data on the Web"



A tree (or graph), unordered (or ordered). With labels on the edges.

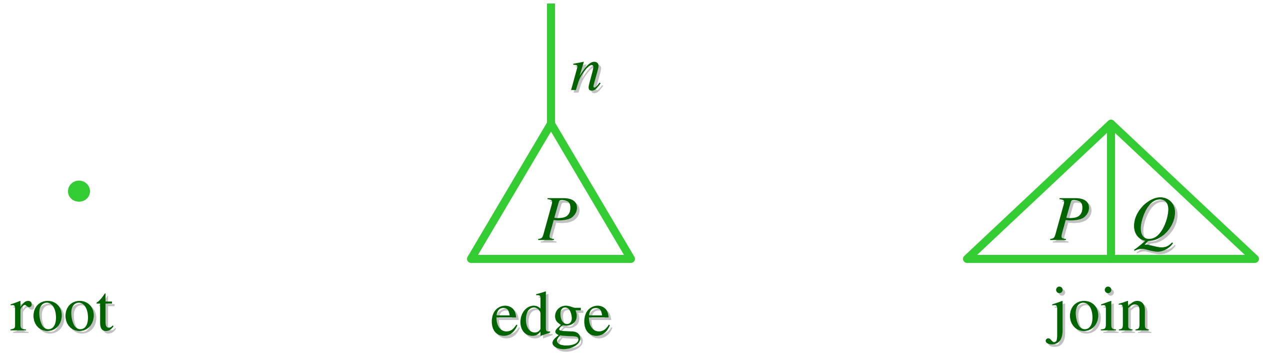
Invented for "flexible" data representation, for quasi-regular data like address books and bibliographies.

Adopted by the DB community as a solution to the "database merge" problem: merging databases from uncoordinated (web) sources.

Adopted by W3C as "web data", then by everybody else.

Trees and their Descriptions

Trees



Syntax for Trees (P,Q)

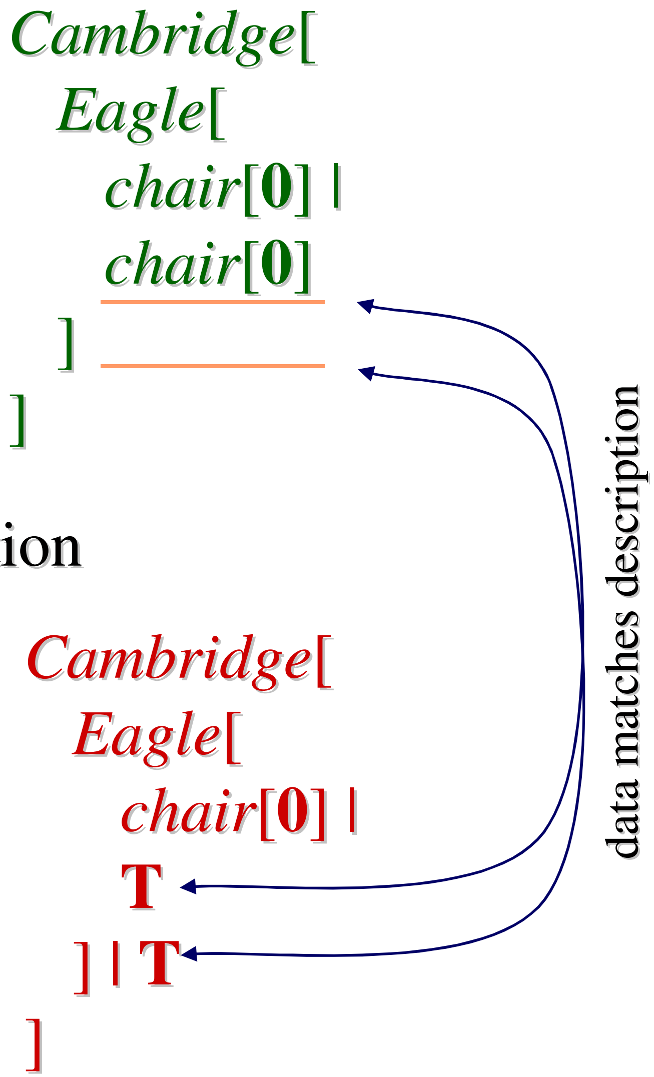
$\mathbf{0}$	root
$n[P]$	edge
$P \mid Q$	join

Basic Descriptions ($\mathcal{A}$,$\mathcal{B}$)

$\mathbf{0}$	there is only a root
$n[\mathcal{A}]$	there is an edge n to a subtree
$\mathcal{A} \mid \mathcal{B}$	there are two joined trees
$\mathbf{T}$	there is anything

Example

Data



```
Cambridge[
  Eagle[
    chair[0] |
    chair[0]
  ]
]
```

In Cambridge there is (nothing but) a pub called the Eagle that contains (nothing but) two empty chairs.

Description

```
Cambridge[
  Eagle[
    chair[0] |
    T
  ] | T
]
```

In Cambridge there is (at least) a pub called the Eagle that contains (at least) one empty chair.

Basic Tree Descriptions

$\mathbf{0}$ matches $\mathbf{0}$

$n[P]$ matches $n[\mathcal{A}]$ iff P matches $\mathcal{A}$

$P \mid Q$ matches $\mathcal{A} \mid \mathcal{B}$ iff P matches $\mathcal{A}$ and Q matches $\mathcal{B}$

P matches $\mathbf{T}$ always

Property: if P matches $\mathcal{A}$ and $P \equiv Q$, then Q matches $\mathcal{A}$

where $P \equiv Q$ means that P and Q represent the same tree:

$$P_1 \mid P_2 \equiv P_2 \mid P_1$$

$$P_1 \mid (P_2 \mid P_3) \equiv (P_1 \mid P_2) \mid P_3$$

$$P \mid \mathbf{0} \equiv P$$

Propositional Descriptions

P matches $\neg\mathcal{A}$ iff P does not match $\mathcal{A}$

P matches $\mathcal{A} \wedge \mathcal{B}$ iff P matches $\mathcal{A}$ and P matches $\mathcal{B}$

P matches $\mathcal{A} \vee \mathcal{B}$ iff P matches $\mathcal{A}$ or P matches $\mathcal{B}$

P matches $\mathcal{A} \Rightarrow \mathcal{B}$ iff if P matches $\mathcal{A}$ then P matches $\mathcal{B}$

Examples

“Vertical” implications about nesting

“Business Policy”

```
Borders[
   Starbucks[…] |
   Books[…]
]
```

$$\mathit{Borders}[\mathbf{T}] \Rightarrow$$
$$\mathit{Borders}[\mathit{Starbucks}[\mathbf{T}] \mid \mathbf{T}]$$

If it’s a Borders,
then it must contain a Starbucks

“Horizontal” implications about proximity

“Social Policy”

```
Smoker[…] |
NonSmoker[…] |
Smoker[…]
```

$$(\mathit{NonSmoker}[\mathbf{T}] \mid \mathbf{T}) \Rightarrow$$
$$(\mathit{Smoker}[\mathbf{T}] \mid \mathbf{T})$$

If there is a NonSmoker,
then there must be a Smoker nearby

Other Descriptions

P matches $\forall x.\mathcal{A}$ iff P matches $\mathcal{A}$ for any label n for x

P matches $\mu X.\mathcal{A}$ iff P matches $\mathcal{A}$ where X is $\mu X.\mathcal{A}$

(up to some well-formedness conditions)

Some definable descriptions:

$$\exists x.\, \mathcal{A} \triangleq \neg\forall x.\neg\mathcal{A}$$

$$\mathit{somewhere}\ \mathcal{A} \triangleq \mu X.\, \mathcal{A} \vee \exists y.\, y[X] \mid \mathbf{T}$$

Descriptions as Schemas

Descriptions are a “very rich type system”. We can comfortably represent various kinds of schemas.

Ex.: Xduce-like schemas (*c.f.* XML DTDs):

$\mathbf{0}$	the empty tree	
$\mathcal{A} \mid \mathcal{B}$	an $\mathcal{A}$ next to a $\mathcal{B}$	
$\mathcal{A} \vee \mathcal{B}$	either an $\mathcal{A}$ or a $\mathcal{B}$	
$n[\mathcal{A}]$	an edge n leading to an $\mathcal{A}$	
$\mathcal{A}^*$	$\triangleq\ \mu X.\mathbf{0} \vee (\mathcal{A} \mid X)$	the merge of zero or more $\mathcal{A}$s
$\mathcal{A}^+$	$\triangleq\ \mathcal{A} \mid \mathcal{A}^*$	the merge of one or more $\mathcal{A}$s
$\mathcal{A}^?$	$\triangleq\ \mathbf{0} \vee \mathcal{A}$	zero or one $\mathcal{A}$

Descriptions as Queries

Yes/no: *Is there an empty chair at the Eagle*?

Eagle[
chair[*John*[**0**]] |
chair[*Mary*[**0**]] |
chair[**0**]
]

matches?

Eagle[
chair[**0**] |
T
]

(Yes)

With match variables $\mathcal{X}$: *Who is sitting at the Eagle?*

matches?

Eagle[
chair[$\mathcal{X}$] |
T
]

Yes: $\mathcal{X}$ = *John*[**0**]
Yes: $\mathcal{X}$ = *Mary*[**0**]
Yes: $\mathcal{X}$ = **0** oops!

With match variables $\mathcal{X}$: *Who is really sitting at the Eagle?*

matches?

Eagle[
  chair[$\neg\mathbf{0} \wedge \mathcal{X}$] |
  T
]

Yes: $\mathcal{X} = \mathit{John}[\mathbf{0}]$

Yes: $\mathcal{X} = \mathit{Mary}[\mathbf{0}]$

With *select-from*:

from Eagle[...]
match Eagle[*chair*[$\neg\mathbf{0} \wedge \mathcal{X}$] | **T**]
select person[$\mathcal{X}$]

Single result:
  person[*John*[**0**]] |
  person[*Mary*[**0**]]

2: A Logic for Graphs

Semistructured data is not actually trees, but either graphs, or graphs with tree backbones.

It is interesting to generalize to a logic for graph.

But this is actually *much* harder. Questions of expressiveness and, particularly, how to define query languages, are still open.

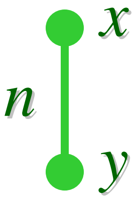
Likely, this is a subset of Second-Order Monadic logic,
but it allows direct “local” reasoning about subgraphs, without explicit disjointness assumptions.

Cardelli-Gardner-Ghelli: ***A Spatial Logic for Querying Graphs***. ICALP’02.

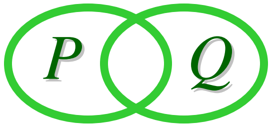
Graphs and their Descriptions

labeled graphs

empty



edge



join

(identify common nodes)

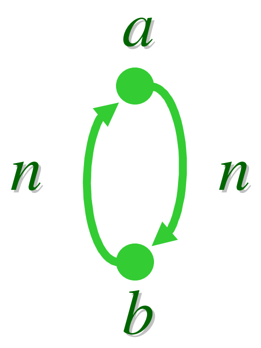
Syntax for Graphs (P,Q)

0	empty
$n(x,y)$	edge
$P \mid Q$	join

Basic Descriptions ($\mathcal{A}$,$\mathcal{B}$)

0	there is an empty graph
$n(x,y)$	there is an edge from x to y labeled n
$\mathcal{A} \mid \mathcal{B}$	there are two disjoint parts of a graph
T	there is anything

Example: Paths



$n(a,b) \mid n(b,a)$

There is a path of length 3 (Yes!):

$$\exists w,x,y,z.\ (n(w,x) \mid \mathbf{T}) \wedge (n(x,y) \mid \mathbf{T}) \wedge (n(y,z) \mid \mathbf{T})$$

There is a non-repeating path of length 3 (No!):

$$\exists w,x,y,z.\ n(w,x) \mid n(x,y) \mid n(y,z) \mid \mathbf{T}$$

As shown in the second case, we can encode implicitly and compactly disjointness assumptions about subgraphs. This is due to the spatial nature of the logic: a formula holds at a subgraph; $\mathcal{A} \mid \mathcal{B}$ means that $\mathcal{A}$ and $\mathcal{B}$ hold at distinct subgraphs. (A “subgraph” here is a subset of edges, not of nodes.)

3: A Logic for Mobility

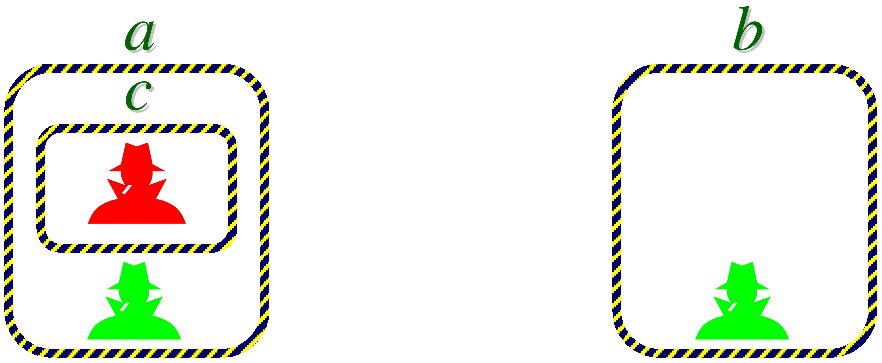
We now look at our original spatial logic: a logic for mobility.

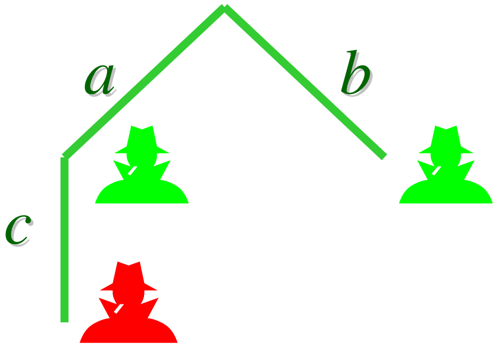
Security specifications are a concern here: additional logical operators arise naturally, and turn out to be logical adjuncts.

Cardelli-Gordon: ***Anytime, Anywhere: Modal Logics for Mobile Ambients***: POPL’00.

Mobility

Mobility is change of spatial structures over time.

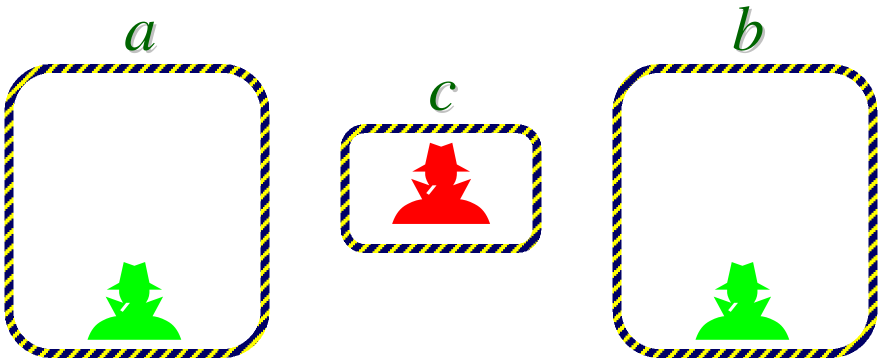


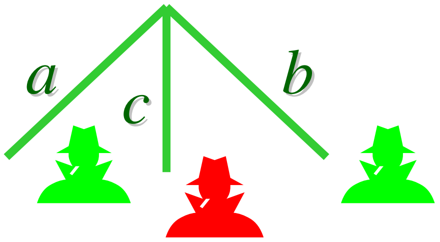


$$a[Q \mid c[\mathit{out}\ a.\ \mathit{in}\ b.\ P]] \qquad \mid b[R]$$

Mobility

Mobility is change of spatial structures over time.

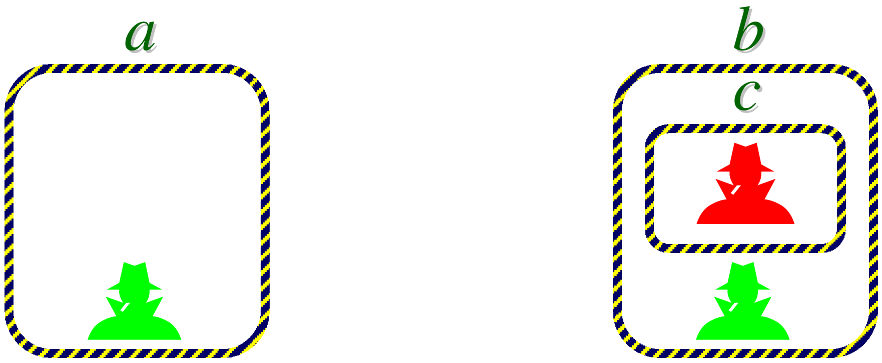


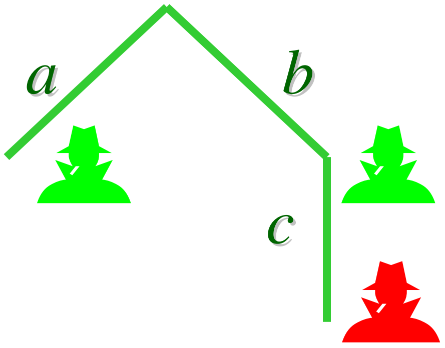


$a[Q] \qquad\qquad \mid c[\mathit{in}\ b.\ P] \ \mid b[R]$

Mobility

Mobility is change of spatial structures over time.



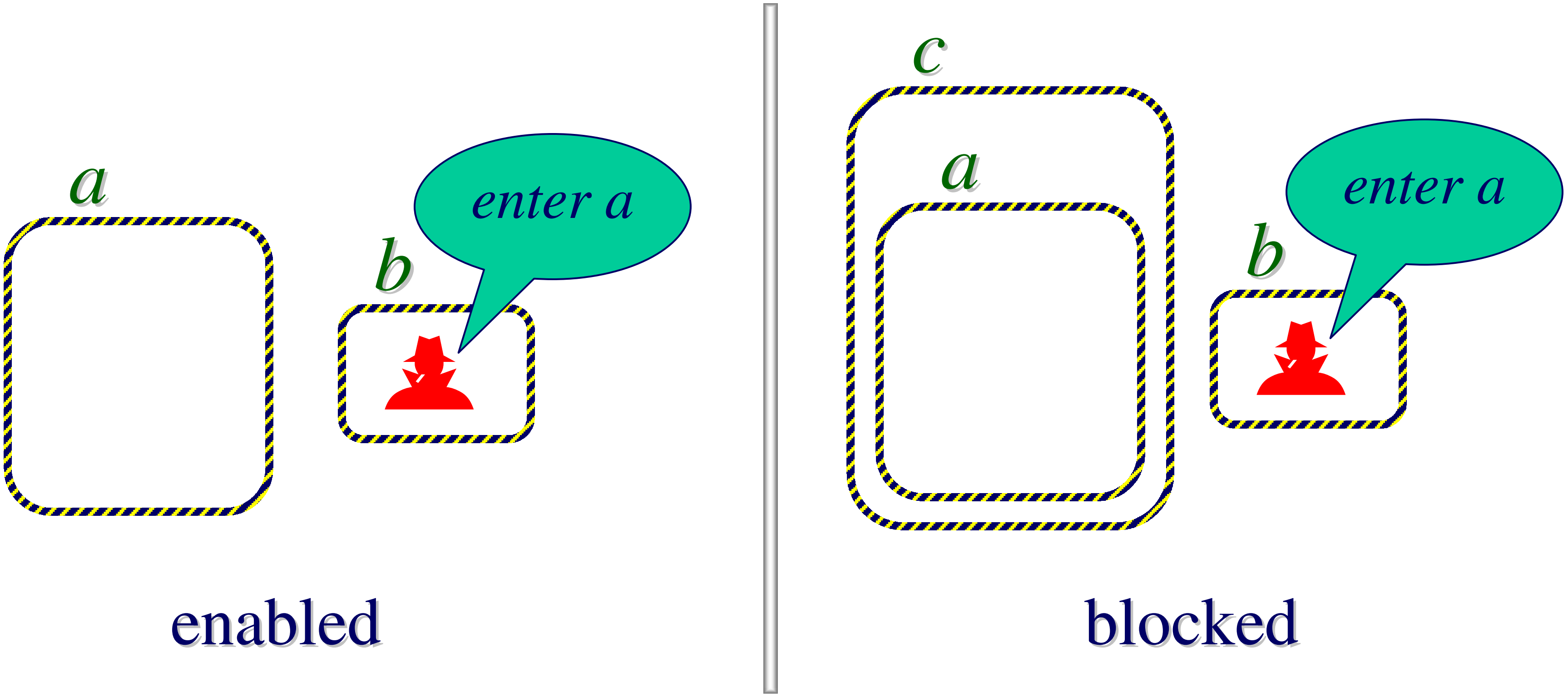


$a[Q] \qquad\qquad \mid b[R \mid c[P]]$

Security

Security issues are reduced to the capability of entering and exiting locations (as well as creating and destroying locations).

- E.g.: Firewall-crossing protocols.
- Capabilities can be exercised only the the right places:



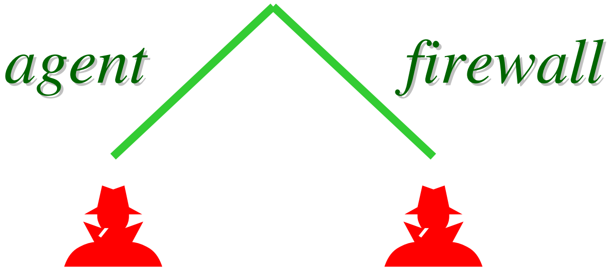
enabled

blocked

Spatial-style Protocol Specification

Right now, we have a spatial configuration, and later, we have another spatial configuration.

E.g.: Right now, the agent is outside the firewall, …

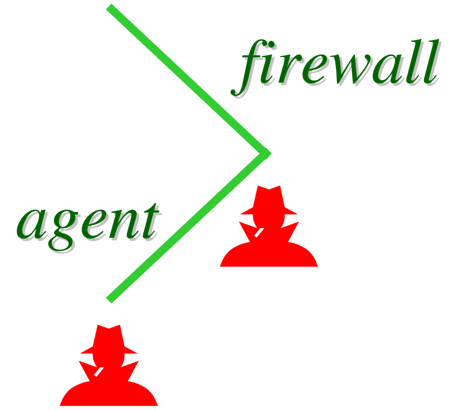


(*agent*[**T**] | *firewall*[**T**] | **T**)

Spatial-style Protocol Specification

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.

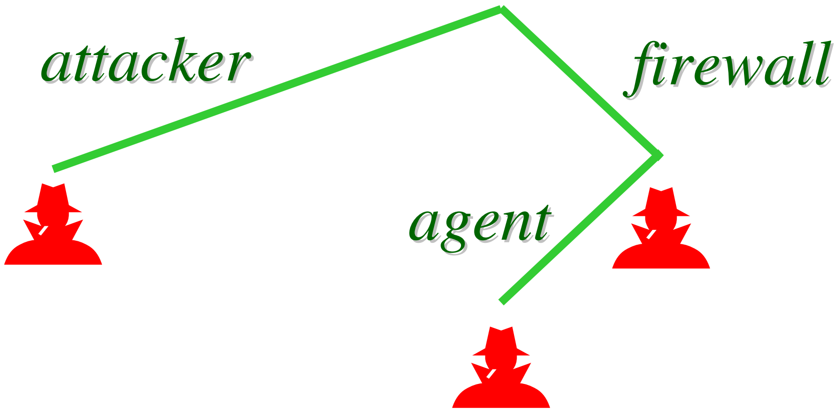


$$(\mathit{agent}[\mathbf{T}] \mid \mathit{firewall}[\mathbf{T}] \mid \mathbf{T}) \land \Diamond(\mathit{firewall}[\mathit{agent}[\mathbf{T}] \mid \mathbf{T}] \mid \mathbf{T})$$

Spatial-style Protocol Specification

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. And this works in presence of any (reasonable) attacker.



$$\mathit{Attack} \triangleright ((\mathit{agent}[\mathbf{T}] \mid \mathit{firewall}[\mathbf{T}] \mid \mathbf{T}) \wedge \Diamond(\mathit{firewall}[\mathit{agent}[\mathbf{T}] \mid \mathbf{T}] \mid \mathbf{T}))$$

Ambient Calculus (without Restriction)

$P \in \Pi$ ::=	Processes		M ::=	Messages
$\mathbf{0}$	void	*spatial*	n	name
$P \mid P'$	composition		$in\ M$	entry capability
$!P$	replication		$out\ M$	exit capability
$M[P]$	ambient		$open\ M$	open capability
$M.P$	capability	*temporal*	ε	empty path
$(n).P$	input		$M.M'$	composite path
$\langle M \rangle$	output			

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

$$M \quad \triangleq \quad M.\mathbf{0}$$ (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.
- 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:

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

- Any details about reduction are "hidden" in the logic within a temporal modality.

Logical Formulas

$\mathcal{A},\mathcal{B} \in \Phi ::=$	Formulas	(η is a name n or a variable x)
$\mathbf{F}$	false	
$\mathcal{A} \wedge \mathcal{B}$	conjunction	
$\mathcal{A} \Rightarrow \mathcal{B}$	implication (adjunct to $\wedge$)	
$\mathbf{0}$	void	
$\mathcal{A} \mid \mathcal{B}$	composition	
$\mathcal{A} \triangleright \mathcal{B}$	guarantee (adjunct to $\mid$)	
$\eta[\mathcal{A}]$	location	
$\mathcal{A}@\eta$	placement (adjunct to $\eta[]$)	
✧$\mathcal{A}$	somewhere	
$\Diamond\mathcal{A}$	sometime	
$\forall x.\mathcal{A}$	quantification over names	

Satisfaction Relation

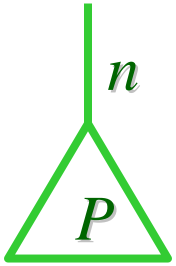
$P \models \mathbf{F}$	*never*	
$P \models \mathcal{A} \wedge \mathcal{B}$	$\triangleq$	$P \models \mathcal{A} \wedge P \models \mathcal{B}$
$P \models \mathcal{A} \Rightarrow \mathcal{B}$	$\triangleq$	$P \models \mathcal{A} \Rightarrow P \models \mathcal{B}$
$P \models \mathbf{0}$	$\triangleq$	$P \equiv \mathbf{0}$
$P \models \mathcal{A} \mid \mathcal{B}$	$\triangleq$	$\exists P', P'' \in \Pi.\ P \equiv P' \mid P'' \wedge P' \models \mathcal{A} \wedge P'' \models \mathcal{B}$
$P \models \mathcal{A} \triangleright \mathcal{B}$	$\triangleq$	$\forall P' \in \Pi.\ P' \models \mathcal{A} \Rightarrow P \mid P' \models \mathcal{B}$
$P \models n[\mathcal{A}]$	$\triangleq$	$\exists P' \in \Pi.\ P \equiv n[P'] \wedge P' \models \mathcal{A}$
$P \models \mathcal{A}@n$	$\triangleq$	$n[P] \models \mathcal{A}$
$P \models$ ✧$\mathcal{A}$	$\triangleq$	$\exists P' \in \Pi.\ P \downarrow^{*} P' \wedge P' \models \mathcal{A}$
$P \models \Diamond \mathcal{A}$	$\triangleq$	$\exists P' \in \Pi.\ P \longrightarrow^{*} P' \wedge P' \models \mathcal{A}$
$P \models \forall x.\mathcal{A}$	$\triangleq$	$\forall m \in \Lambda.\ P \models \mathcal{A}\{x \leftarrow m\}$

$P \downarrow P'$ iff $\exists n, P''.\ P \equiv n[P'] \mid P''$

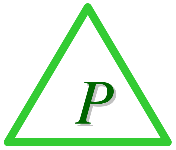
$\downarrow^{*}$ is the reflexive and transitive closure of $\downarrow$

Satisfaction for Basic Operators

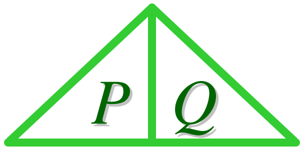
- $\models \mathbf{0}$



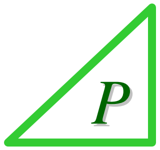
$\models n[\mathcal{A}]$ if



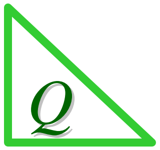
$\models \mathcal{A}$



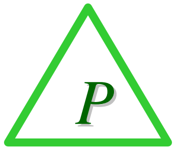
$\models \mathcal{A} \mid \mathcal{B}$ if



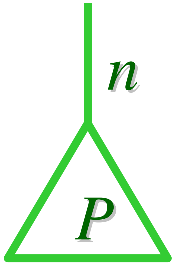
$\models \mathcal{A}$ and



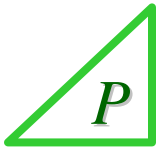
$\models \mathcal{B}$



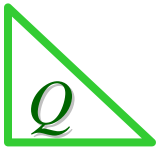
$\models \mathcal{A}@n$ if



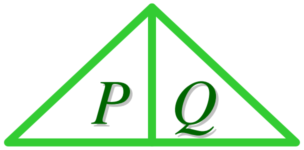
$\models \mathcal{A}$



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

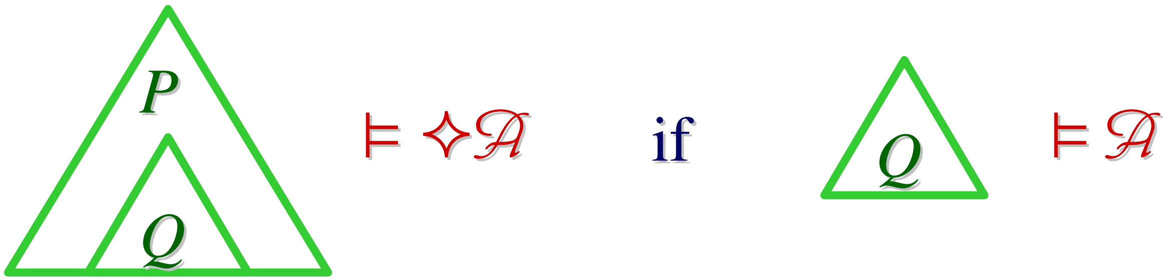


$\models \mathcal{A}$ we have

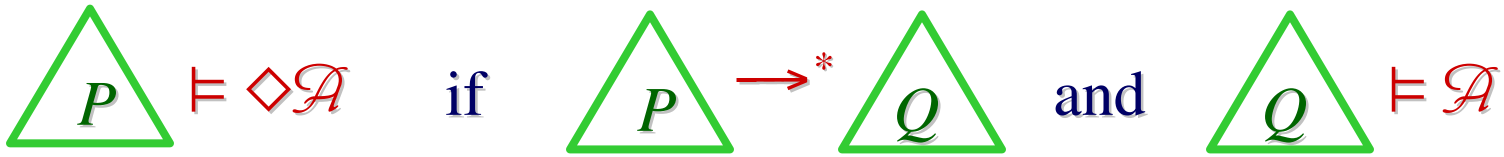


$\models \mathcal{B}$

Satisfaction for Somewhere/Sometime



P Q $\models$ ✧$\mathscr{A}$ if Q $\models \mathscr{A}$



P $\models \Diamond\mathscr{A}$ if $P \longrightarrow^{*} Q$ and Q $\models \mathscr{A}$

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

Basic Fact: satisfaction is invariant under structural congruence:

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

Hence, formulas describe only congruence-invariant properties.

Applications

Verifying security+mobility protocols.

- Still hard, but we have good techniques. [Gordon-Cardelli '99]

Modelchecking security+mobility assertions:

- If P is $\mathbf{!}$-free and $\mathcal{A}$ is $\triangleright$-free, then $P \models \mathcal{A}$ is decidable. (PSPACE-complete [Cheratonik et al. '01].)
- If P is $\mathbf{!}$-free and $\mathcal{A}$ is $\forall$-free, then $P \models \mathcal{A}$ is decidable. [Cardelli-Calcagno-Gordon '02].)
- These provides ways 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.

4: New Logics for Concurrency

In the process of making spatial sense of $n[\mathcal{A}]$, we also had to make spatial sense of $\mathcal{A} \mid \mathcal{B}$. The latter is, in fact, the harder part. So, in retrospect, it makes sense to consider it on its own.

An outcome is spatial logics for CCS/CSP-like process calculi. Basic idea: take a Hennessy-Milner modal logic and add an $\mathcal{A} \mid \mathcal{B}$ operator. ([Dam] Very hard to reconcile with bisimulation.)

One can go further and investigate spatial logics for restriction, with a *hiding quantifier* $\mathsf{H}x.\mathcal{A}$ (e.g. for π-calculus). This is essential for security/privacy specifications.
([Caires] Very hard to reconcile with bisimulation.)

We can make all that work smoothly by taking a very *intensional* point of view. The logical formulas are not *up-to-bisimulation*: they are *up-to-structural-congruence*.

Spatial Properties: Identifiable Subsystems

A system is often composed of identifiable subsystems.

- “A message is sent from Alice to Bob.”
- “The protocol is split between two participants.”
- “The virus attacks the server.”

Such partitions of a system are (obviously) spatial properties. They correspond to a spatial arrangement of processes in different places.

- Process calculi are *very* good at expressing such arrangements operationally (*c.f.*, chemical semantics, structural congruence).
- To the point where a process is often used as a specification of another process! (We consider this as an anomaly.)
- We want something equally good at the specification, or logical, level.

Spatial Properties: Restricted Resources

A system often restricts the use of certain resources to certain subsystems.

- "A shared private key *n* is established between two processes."
- "A fresh nonce *n* is generated locally and transmitted."
- "The applet runs in a secret sandbox."

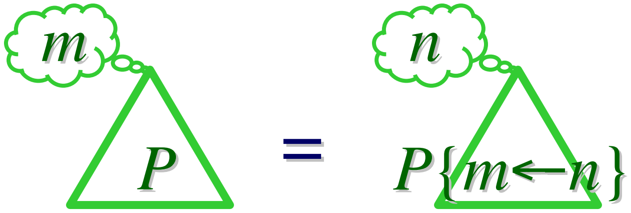
Something is *hidden/secret/private* if it is present only in a limited subsystem. So these are spatial properties too.

- If something is secret, by assumption it cannot be known. Still, we want to talk about it in specifications.
- We can talk about a secret name only by using a *fresh* name for it (we cannot assume the secret name matches any known name).
- So freshness will be an important concept. Logics of freshness are very new.

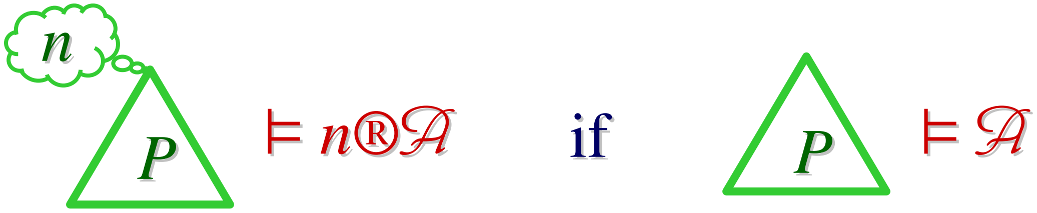
Typical Spatial Formulas for Concurrency

Processes		*Formulas*	
$\mathbf{0}$	(void)	$\mathbf{0}$	(nothing here)
$P \mid Q$	(composition)	$\mathcal{A} \mid \mathcal{B}$	(two things here)
$(\nu n)P$	(restriction)	$n \circledR \mathcal{A}$	(hidden thing here)
$n\langle m\rangle$	(message)	$n\langle m\rangle$	(a message here)

$$n\langle m\rangle \vDash n\langle m\rangle$$



+ Scope Extrusion…



Things one can say

Single-threaded (or void):

$\neg(\neg\mathbf{0} \mid \neg\mathbf{0})$

Output: outputs a message m on n (and is/does nothing else):

$n\langle m\rangle$

In presence of a message m on n, sends a message n on m and stops:

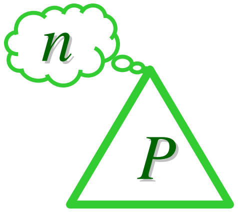
$n\langle m\rangle \triangleright \text{»}m\langle n\rangle$

Contains a name free:

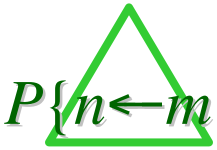
$\copyright n \quad \triangleq \quad \neg n \circledR \mathbf{T}$

$P \vDash \neg n \circledR \mathbf{T}$ iff $\neg\ P \equiv (\nu n)P'$ iff $n \in fn(P)$

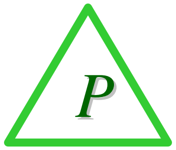
Satisfaction for Hidden and Public Names



$\models$ H$x.\mathcal{A}$ if $\exists m \notin fn(P,\mathcal{A})$



$P\{n \leftarrow m\} \models \mathcal{A}\{x \leftarrow m\}$



$\models$ ©n if $n \in fn(P)$

(Technically, H$x.\mathcal{A}$ and ©n are both defined from n®$\mathcal{A}$ and a Gabbay-Pitts freshness quantifier.)

Example: “Shared Secret” Postcondition

Consider a situation where “a hidden name x is shared by two locations n and m, and is not known outside those locations”.

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

What can we do with such a spec? We can fully expand the definitions and work it out in the process calculus:

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

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

- E.g.: take $P = (\nu p)\ (n[p[]] \mid m[p[]])$.

Or we can work logically at the formula level, within a proof system.

Ex: Immovable Object vs. Irresistible Force

$$\mathit{Im} \triangleq \mathbf{T} \triangleright \Box(\mathit{obj}\langle\rangle \mid \mathbf{T})$$

$$\mathit{Ir} \triangleq \mathbf{T} \triangleright \Box\Diamond\neg(\mathit{obj}\langle\rangle \mid \mathbf{T})$$

$$\mathit{Im} \mid \mathit{Ir} \vdash (\mathbf{T} \triangleright \Box(\mathit{obj}\langle\rangle \mid \mathbf{T})) \mid \mathbf{T} \qquad \mathcal{A} \vdash \mathbf{T}$$

$$\vdash \Box(\mathit{obj}\langle\rangle \mid \mathbf{T}) \qquad (\mathcal{A} \triangleright \mathcal{B}) \mid \mathcal{A} \vdash \mathcal{B}$$

$$\vdash \Diamond\Box(\mathit{obj}\langle\rangle \mid \mathbf{T}) \qquad \mathcal{A} \vdash \Diamond\mathcal{A}$$

$$\mathit{Im} \mid \mathit{Ir} \vdash \mathbf{T} \mid (\mathbf{T} \triangleright \Box\Diamond\neg(\mathit{obj}\langle\rangle \mid \mathbf{T})) \qquad \mathcal{A} \vdash \mathbf{T}$$

$$\vdash \Box\Diamond\neg(\mathit{obj}\langle\rangle \mid \mathbf{T}) \qquad \Diamond\neg\mathcal{A} \vdash \neg\Box\mathcal{A}$$

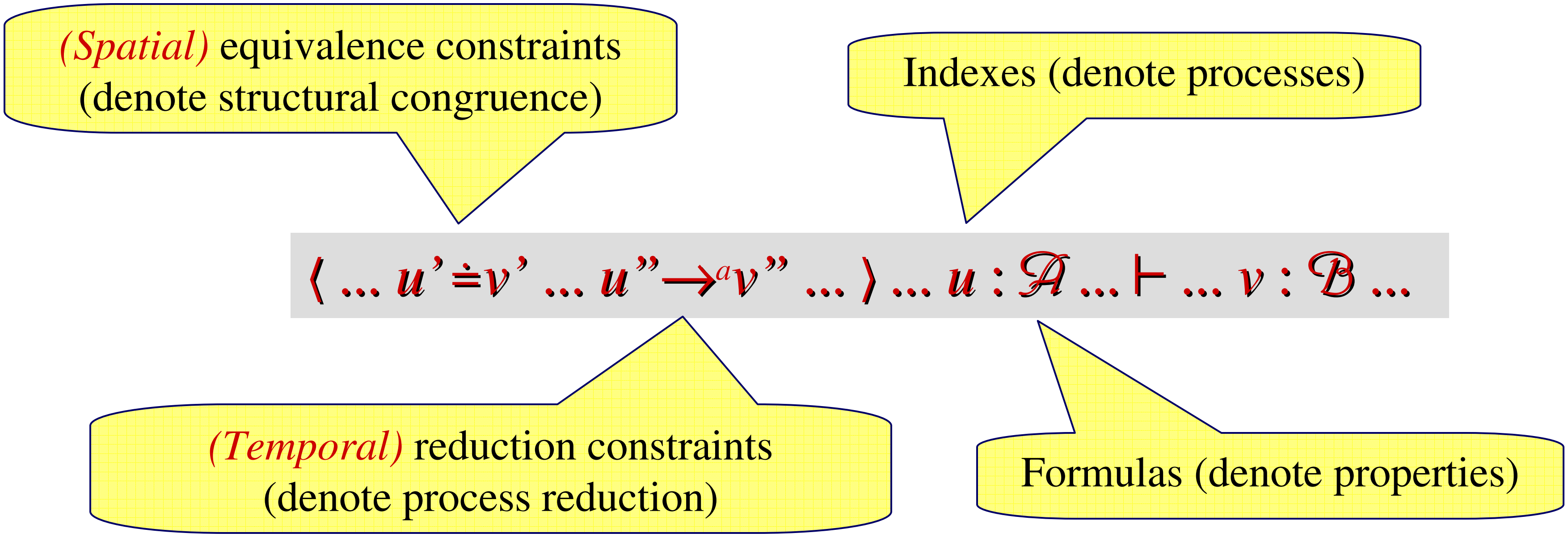
$$\vdash \neg\Diamond\Box(\mathit{obj}\langle\rangle \mid \mathbf{T}) \qquad \Box\neg\mathcal{A} \vdash \neg\Diamond\mathcal{A}$$

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

5: A Common Formalization Style

Many-world sequents: $\langle S \rangle\, \Gamma \vdash \Delta$

Validity: if all the constraints S_k and all the assumptions Γ_i are satisfied, then one of the conclusions Δ_j is satisfied



(Spatial) equivalence constraints (denote structural congruence)

Indexes (denote processes)

$\langle \ldots u' \doteq v' \ldots u'' \rightarrow^{a} v'' \ldots \rangle \ldots u : \mathcal{A} \ldots \vdash \ldots v : \mathcal{B} \ldots$

(Temporal) reduction constraints (denote process reduction)

Formulas (denote properties)

Alex Simpson did this for temporal logic.

Caires-Cardelli: A Spatial Logic for Concurrency (Part II).

Recipe for Rules

- *Left rules*, *right rules*. Operate mainly on the $\Gamma \vdash \Delta$ part.
  When operating on constraints $\langle S \rangle$:

  Going up: One adds, the other checks constraints.

  Going down: One removes, the other assumes constraints.

  They form cut elimination pairs.

- *World rules (optional)*. Operate on the $\langle S \rangle$ part only.
  Embody inversion lemmas: deep properties of process calculi.
  (In temporal logic, they embody properties such as reflexivity and transitivity of the reachibility relation.)

  Going up: add deducible constraints.

  Going down: remove redundant constraints.

  Commute easily with cuts.

Propositional Connectives

Identity, Cut, and Contraction:

(Id)

$$\frac{u \doteq_S u' \quad \mathcal{A} \equiv_S \mathcal{A}'}{\langle S \rangle\, \Gamma, u : \mathcal{A} \vdash u' : \mathcal{A}', \Delta}$$

(Cut)

$$\frac{\langle S \rangle\, \Gamma \vdash u : \mathcal{A}, \Delta \quad \langle S \rangle\, \Gamma, u : \mathcal{A} \vdash \Delta}{\langle S \rangle\, \Gamma \vdash \Delta}$$

(C L)

$$\frac{\langle S \rangle\, \Gamma, u : \mathcal{A}, u : \mathcal{A} \vdash \Delta}{\langle S \rangle\, \Gamma, u : \mathcal{A} \vdash \Delta}$$

(C R)

$$\frac{\langle S \rangle\, \Gamma \vdash u : \mathcal{A}, u : \mathcal{A}, \Delta}{\langle S \rangle\, \Gamma \vdash u : \mathcal{A}, \Delta}$$

Propositional Connectives:

($\wedge$ L)

$$\frac{\langle S \rangle\, \Gamma, u : \mathcal{A}, u : \mathcal{B} \vdash \Delta}{\langle S \rangle\, \Gamma, u : \mathcal{A} \wedge \mathcal{B} \vdash \Delta}$$

($\wedge$ R)

$$\frac{\langle S \rangle\, \Gamma \vdash u : \mathcal{A}, \Delta \quad \langle S \rangle\, \Gamma \vdash u : \mathcal{B}, \Delta}{\langle S \rangle\, \Gamma \vdash u : \mathcal{A} \wedge \mathcal{B}, \Delta}$$

($\Rightarrow$ L)

$$\frac{\langle S \rangle\, \Gamma \vdash u : \mathcal{A}, \Delta \quad \langle S \rangle\, \Gamma, u : \mathcal{B} \vdash \Delta}{\langle S \rangle\, \Gamma, u : \mathcal{A} \Rightarrow \mathcal{B} \vdash \Delta}$$

($\Rightarrow$ R)

$$\frac{\langle S \rangle\, \Gamma, u : \mathcal{A} \vdash u : \mathcal{B}, \Delta}{\langle S \rangle\, \Gamma \vdash u : \mathcal{A} \Rightarrow \mathcal{B}, \Delta}$$

(F L)

$$\frac{}{\langle S \rangle\, \Gamma, u : \mathbf{F} \vdash \Delta}$$

(F R)

$$\frac{\langle S \rangle\, \Gamma \vdash \Delta}{\langle S \rangle\, \Gamma \vdash u : \mathbf{F}, \Delta}$$

Spatial Connectives

Composition:

(0 L)

$$\frac{\langle S, u \doteq 0\rangle\, \Gamma \vdash \Delta}{\langle S\rangle\, \Gamma, u : \mathbf{0} \vdash \Delta}$$

(0 R)

$$\frac{u \doteq_S 0}{\langle S\rangle\, \Gamma \vdash u : \mathbf{0}, \Delta}$$

(| L) *$\mathcal{X}, \mathcal{Y}$ not free in the conclusion*

$$\frac{\langle S, u \doteq \mathcal{X} \mid \mathcal{Y}\rangle\, \Gamma, \mathcal{X} : \mathcal{A}, \mathcal{Y} : \mathcal{B} \vdash \Delta}{\langle S\rangle\, \Gamma, u : \mathcal{A} \mid \mathcal{B} \vdash \Delta}$$

(| R)

$$\frac{\langle S\rangle\, \Gamma \vdash v : \mathcal{A}, \Delta \quad \langle S\rangle\, \Gamma \vdash t : \mathcal{B}, \Delta \quad u \doteq_S v|t}{\langle S\rangle\, \Gamma \vdash u : \mathcal{A} \mid \mathcal{B}, \Delta}$$

Guarantee:

(▷ L)

$$\frac{\langle S\rangle\, \Gamma \vdash t : \mathcal{A}, \Delta \quad \langle S\rangle\, \Gamma, t|u : \mathcal{B} \vdash \Delta}{\langle S\rangle\, \Gamma, u : \mathcal{A} \triangleright \mathcal{B} \vdash \Delta}$$

(▷ R) *$\mathcal{X}$ not free in the conclusion*

$$\frac{\langle S\rangle\, \Gamma, \mathcal{X} : \mathcal{A} \vdash v : \mathcal{B}, \Delta \quad v \doteq_S \mathcal{X} | u}{\langle S\rangle\, \Gamma \vdash u : \mathcal{A} \triangleright \mathcal{B}, \Delta}$$

i.e.:

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

Additional World Structure:

(S | 0)

$$\frac{\langle S, u \doteq \mathbf{0}\rangle\, \Gamma \vdash \Delta \quad u|v \doteq_S \mathbf{0}}{\langle S\rangle\, \Gamma \vdash \Delta}$$

(S | |) *$\mathcal{X}, \mathcal{Y}, \mathcal{U}, \mathcal{V}$ not free in the conclusion*

$$\frac{\langle S, u \doteq \mathcal{X} \mid \mathcal{Y}, v \doteq \mathcal{U} \mid \mathcal{V}, t \doteq \mathcal{X} \mid \mathcal{U}, w \doteq \mathcal{Y} \mid \mathcal{V}\rangle\, \Gamma \vdash \Delta \quad u|v \doteq_S t|w}{\langle S\rangle\, \Gamma \vdash \Delta}$$

Suppose $x|y=0 \Rightarrow x=0$. Then, if we can already deduce that $x|y \doteq_S 0$, we can eliminate a redundant assumption $x \doteq 0$.

Suppose $u|v=t|s \Rightarrow \exists\, x,y,z,w$ s.t. $u=x|y$, $v=z|w$, $t=x|z$, $s=y|w$. Then, if we can already deduce that $u|v \doteq_S t|w$, we can eliminate a redundant assumptions

$(\mathcal{A} \mid \mathcal{B}) \wedge \mathbf{0} \vdash \mathcal{A} \wedge \mathcal{B}$

6.2 $\langle S,\ u \doteq \mathcal{X} \mid \mathcal{Y},\ u \doteq \mathbf{0},\ \mathcal{X} \doteq \mathbf{0} \rangle\ \Gamma, \mathcal{X} : \mathcal{A},\ \mathcal{Y} : \mathcal{B} \vdash u : \mathcal{A}, \Delta$	(Id) since $u = \mathcal{X}$
5.2 $\langle S,\ u \doteq \mathcal{X} \mid \mathcal{Y},\ u \doteq \mathbf{0} \rangle\ \Gamma, \mathcal{X} : \mathcal{A},\ \mathcal{Y} : \mathcal{B} \vdash u : \mathcal{A}, \Delta$	6.2, (S | **0**) since $\mathcal{X} \mid \mathcal{Y} = \mathbf{0}$
4.2 $\langle S,\ u \doteq \mathcal{X} \mid \mathcal{Y} \rangle\ \Gamma, \mathcal{X} : \mathcal{A},\ \mathcal{Y} : \mathcal{B},\ u : \mathbf{0} \vdash u : \mathcal{A}, \Delta$	5.2, (**0** L)
3.2 $\langle S \rangle\ \Gamma, u : (\mathcal{A} \mid \mathcal{B}),\ u : \mathbf{0} \vdash u : \mathcal{A}, \Delta$	4.2, (| L)
2.2 $\langle S \rangle\ \Gamma, u : (\mathcal{A} \mid \mathcal{B}) \wedge \mathbf{0} \vdash u : \mathcal{A}, \Delta$	3.2, ($\wedge$ L)
...	
2.1 $\langle S \rangle\ \Gamma, u : (\mathcal{A} \mid \mathcal{B}) \wedge \mathbf{0} \vdash u : \mathcal{B}, \Delta$	Similarly
1 $\langle S \rangle\ \Gamma, u : (\mathcal{A} \mid \mathcal{B}) \wedge \mathbf{0} \vdash u : \mathcal{A} \wedge \mathcal{B}, \Delta$	2.1, 2.2, ($\wedge$ R)

Conclusions

We set out to find logics for describing properties of distributed systems.
(After trying equational reasoning, traces, etc.)

Spatial logics exhibit the trade-offs of temporal logics: compact notation for implicit state, nice proof systems, reduced expressiveness.

Along the way, we discovered many other applications for the basic techniques. We believe there is something intriguing and new in the approach and its formalization.

With respect to traditional logics of concurrency, we are very *intensional*.
But another word for it is *precise*.

With Caires, we now have a logic and sequent calculus (with cut-elimination) for π-calculus, where we can express privacy properties.

Related work:

- With Calcagno and Godon: Model checking and validity checking.
- Sangiorgi: Spacetime bisimulation.
- O'Hearn and Pym: Logics for heaps.