

SemiStructured ~~Mobile~~ Computation

Luca Cardelli

Microsoft Research

Invited talk. Arising from work with
Andrew D. Gordon and Giorgio Ghelli

DBPL'99 Kinloch Rannoch

Stream of Thought

Hmmm... I am invited to talk at DBPL'99. Glad to accept but...

What could I possibly talk about? All I've done recently is mobile computation. Just hope they'll listen to that? Doubtful...

Last time I looked, it was all OODB stuff; maybe I can recycle an OO talk. Better catch up with the literature; I'll ask Giorgio.

What's this semistructured stuff? Looks vaguely familiar. Wait... wait... *That's what I've been doing!!!*

BINGO! I have a talk. And it's the SAME as the mobility talk! (Up to isomorphism.)

Insight

- These endeavors are similar:

Mobile Computation \approx SemiStructured DataBases

because they share the same kind of dynamicity of structure:

"Cannot rely on uniform structure" (SSDB)

\approx "Cannot rely on things staying put" (MC)

- Thesis:

(1) *Mobile computation is the generalization of semistructured data to computation,*

(2) *Semistructured databases are the generalization of mobile computation to bulk data.*

- General thesis: **Mobile Blah \approx SemiStructured Blah**

Thesis Instance: Information

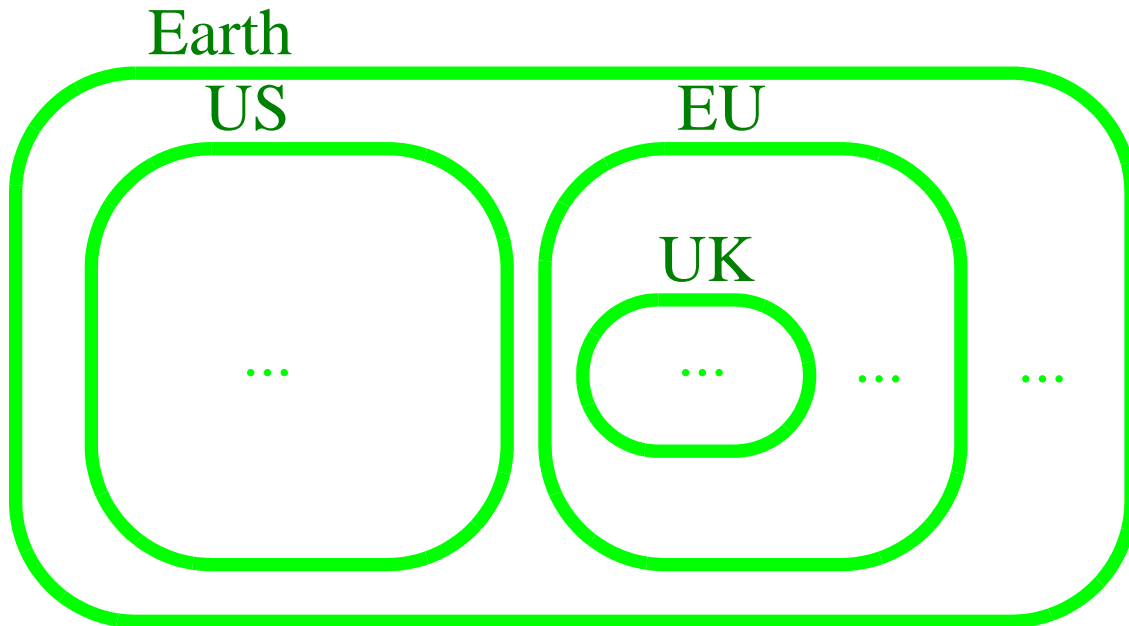
Hmmm... They can't have proper data structures to represent information. No proper records or variants. So they have these funky edge-labeled trees and graphs. Looks pretty awful.

Wait... *edge-labeled trees*?!/? That's what we use to represent information!

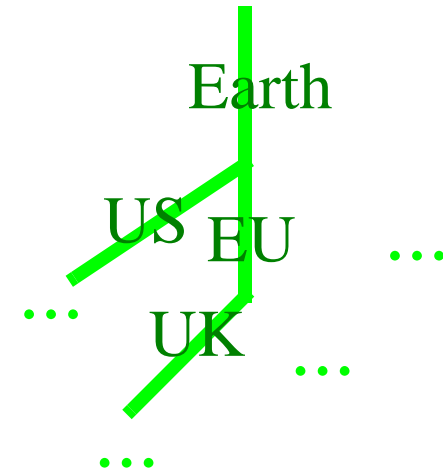
Representation of Spatial Information

- Postulate: space is tree-structured.

Geographical maps



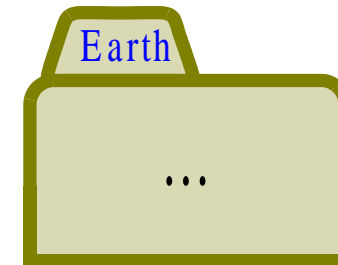
Edge-labeled trees



Expressions

Earth[US[...] | EU[UK[...] | ...] ...]

Folders



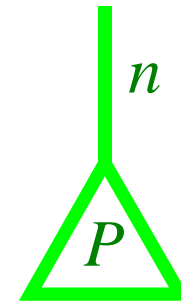
Basic Expressions

- $\mathbf{0}$ is the tree which is just a root

$\mathbf{0}$ represents \blacksquare

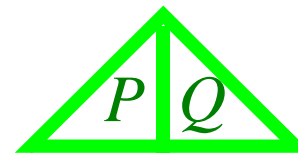
- $n[P]$ is a tree with of a single root edge n , and with descendent P .

$n[P]$ represents



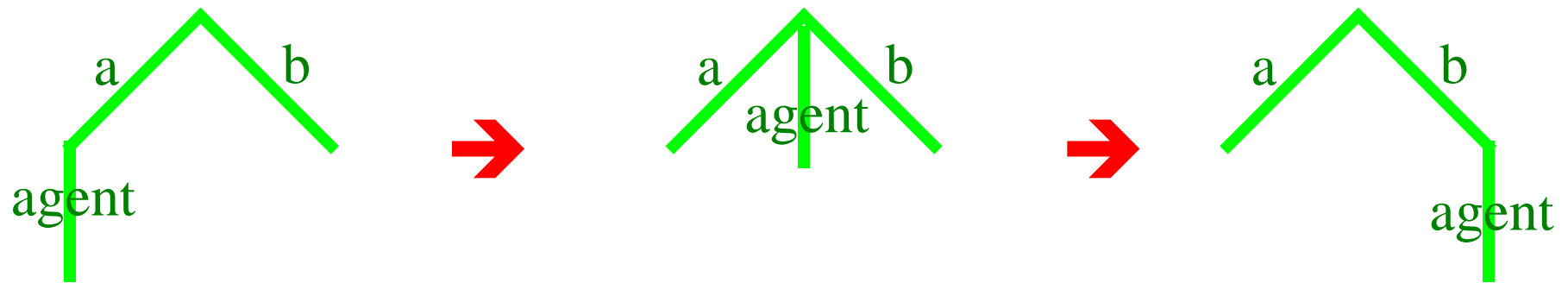
- $P | Q$ is a tree made of two subtrees P and Q joined at the root.

$P | Q$ represents



Mobility

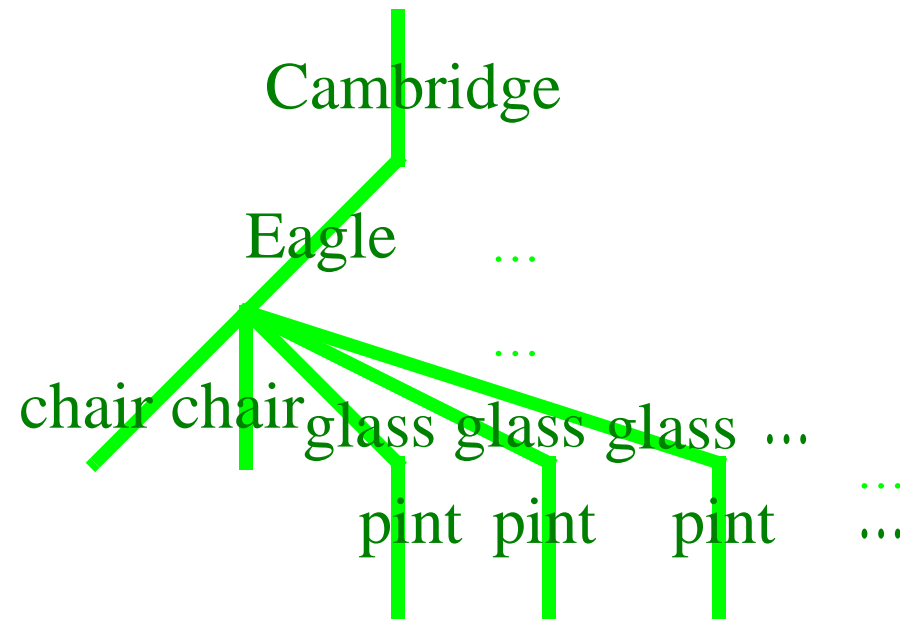
- Then, *mobility* is change of structures over time:



$a[\text{agent}[]] \mid b[] \rightarrow a[] \mid \text{agent}[] \mid b[] \rightarrow a[] \mid b[\text{agent}[]]$

Information Trees

- Our basic model of information is going to be *finite-depth edge-labeled unordered trees*; for short: *info trees*.
- One subtlety: unbounded resources are represented by infinite branching:



Expressions

- We use *expressions* to describe info trees. These are nested expressions with ! for unbounded replication.

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

Cambridge[!ParkingSpace[] | ...] (not!)

- Two spatial expressions are equivalent when they describe the same spatial tree.

– Ex.:

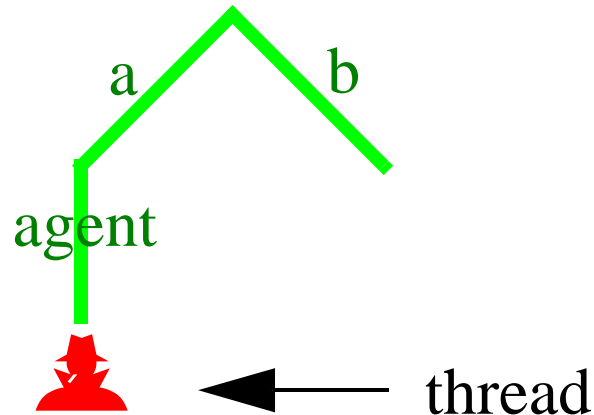
$$a[] | b[] \equiv b[] | a[]$$

$$a[] | !a[] \equiv !a[]$$

- This is not totally trivial (because of !), but we have a complete axiomatization of such equivalence.

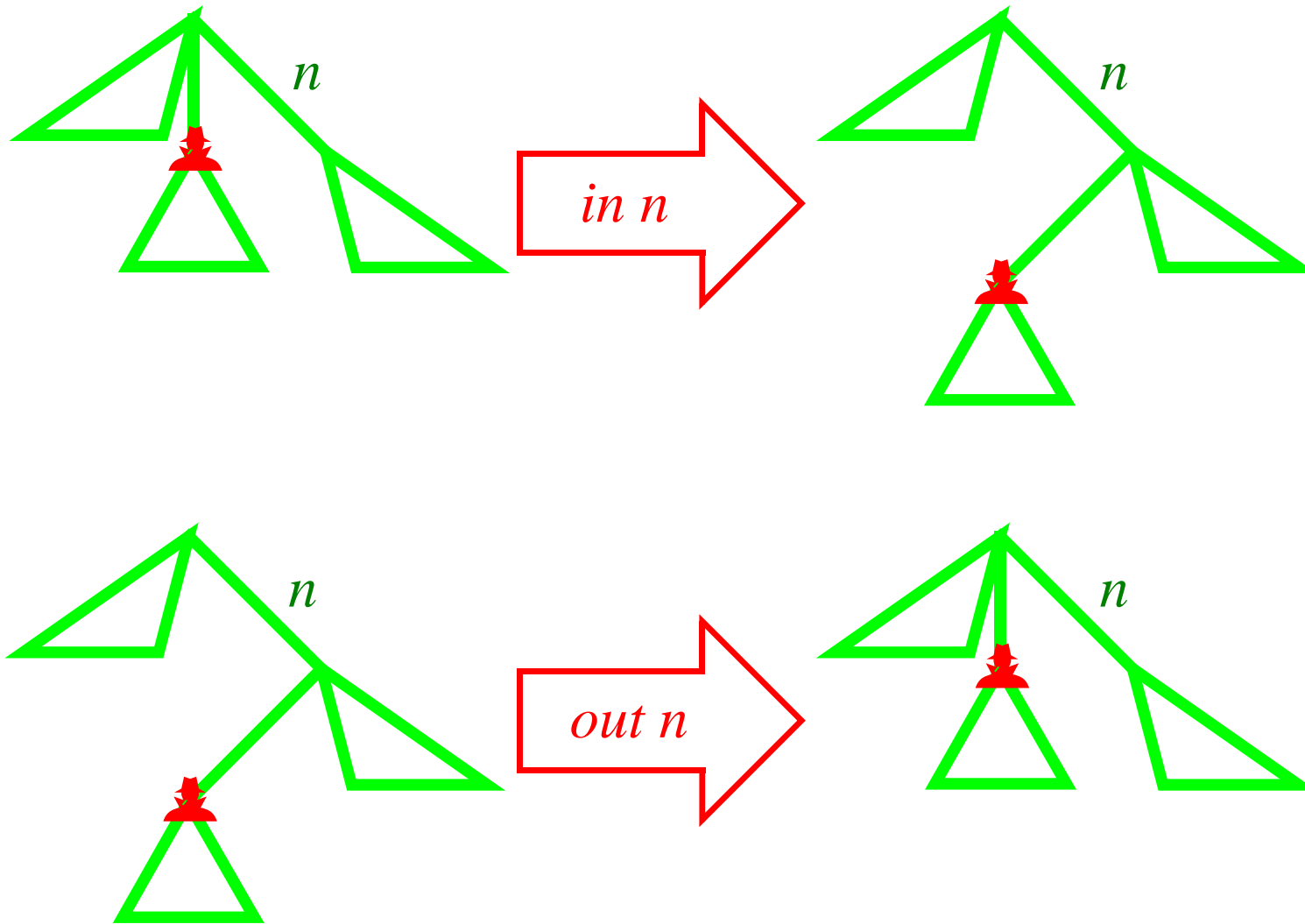
Ambient Expressions

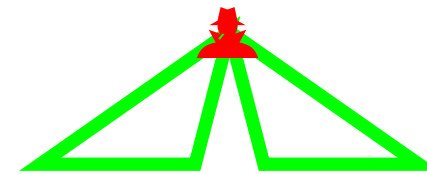
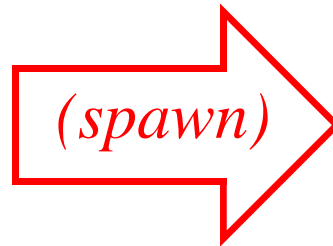
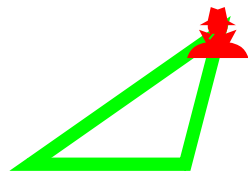
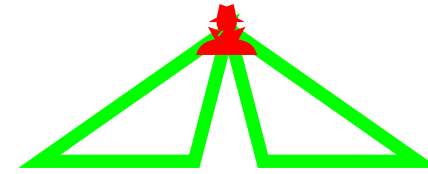
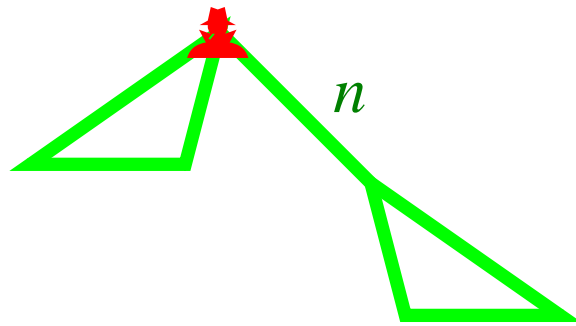
- Info expressions/trees are a subset of *ambient expressions/trees*, where we can represent also the dynamic aspects of (mobile) computation and (mutable) information.



- The ambient calculus has a spatial and a temporal component.
 - The spatial component consists of edge-labeled trees, i.e. semi-structured data.
 - The temporal component includes of operations that locally modify the spatial component.

Ambient Operations





(Turing-complete, together with iteration and name creation.)

Thesis Instance: Representation of Data Structures

Hmmm... They can't rely on records being the same. Bummer.
Wait... we can't rely on records being the same either!

- In the ambient calculus we represent records $\{l_1=v_1, \dots, l_n=v_n\}$ as:

$$r[l_1[\langle v_1 \rangle \dots] \mid \dots \mid l_n[\langle v_n \rangle \dots]]$$

where r is the name (address) of the record, which is used to name an ambient $r[\dots]$. This contains subambients $l_1[\dots] \dots l_n[\dots]$ representing labeled fields (unordered because \mid is too). The field ambients contain the field values v_1, \dots, v_n and some machinery to allow them to be read and rewritten.

- But: ambients are mobile. This means that, potentially, field subambients $l_i[\dots]$ can take off and leave, and new fields can arrive!
- What could be more semistructured?

Thesis Instance: Type Systems

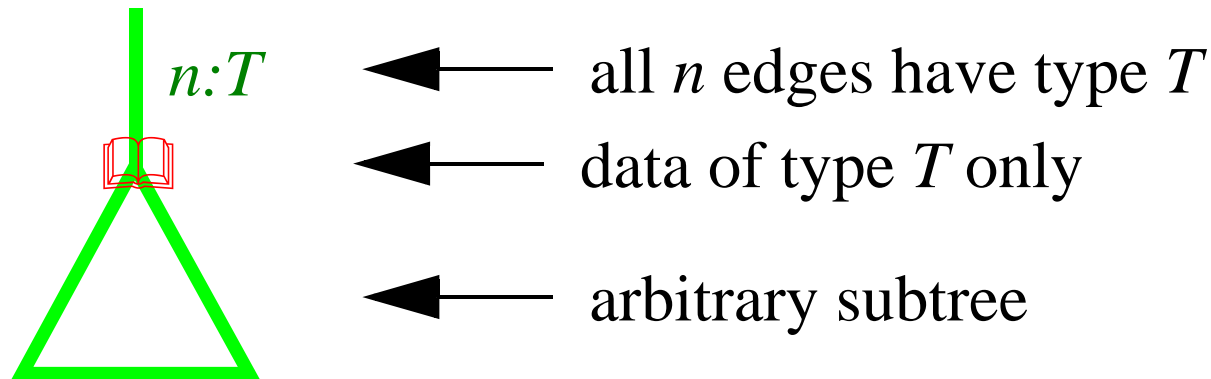
Hmmm... Type systems for semistructured data... there is a black hole! No way it can ever work.

Wait... *we've already done it!*

- We have developed type systems for mobile computation that tolerate dynamicity of structure.
 - These are not weak systems: they can encode standard type systems for λ -calculus and π -calculus. Still:
 - They do not assume uniformity of structure (they can't).
 - They guarantee uniformity of interaction within a dynamic heterogeneous structure.
- Therefore, these are type systems for semistructured data. We have several of them, some of them constraining the degree of semistructuredness ("immobility").

A Type System for SemiStructured Data

- Imagine a typed file system where each named folder:
 - can contain only a single kind of files (based on the folder name)
 - can contain subfolders of any kind



- This is a semistructured type system:
 - Restricts the data that can sit at any node.
 - Does not restrict the structure of the tree. (Unless extended...)
 - N.B.: Properly restricts *open* (subtree merge).

Thesis Instance: Queries

Hmmm... Semistructured data, fine. Semistructured computation, great. But what's a DB without a query language? Without a query logic?

Wait... *logic?!?* We got logic! We got algebra! Hence, (perhaps) we got queries!

Modal Logics

- In standard logic, assertions are either **true** or **false**.
- In a *modal* logic, the truth of an assertion is relative to a *state*.
 - In epistemic logic: a *knowledge state*.
 - In temporal logic: an *execution state*.
 - In our logic: a *space-time state*, relative to the *current place* and the *current time*.
- Here is a **formula** talking about a tree (not a **tree** itself):

Cambridge[Eagle[chair[] | ..] | ..]

Right now in Cambridge there is a pub called the Eagle, and inside the Eagle there is at least an empty chair.

This may be true or false depending on the time of day (happy hour?) and location (Cambridge England or Mass.?).

Satisfaction

- There is a satisfaction relation between a tree and a formula.

$$P \models \mathcal{A}$$

Spatially, this can be understood as a matching process... that is:

- A query.
 - $P \models \mathcal{A}$ means: see if tree P matches the query \mathcal{A} and (for DB applications) return information about the match.
 - By searching for all possible ways in which $P \models \mathcal{A}$ can be satisfied, we obtain a collection of answers.
 - A typical semistructured-like query is: "is there somewhere a subtree that has the shape *such and such*".
- We have a (model-checking-style) algorithm for deciding $P \models \mathcal{A}$ for finite queries. It can be adapted to matching (work in progress...).

Logical Formulas

$\mathcal{A}, \mathcal{B} : \Phi ::=$

\mathbf{T}

true

$\neg \mathcal{A}$

negation

$\mathcal{A} \vee \mathcal{B}$

disjunction

$\mathbf{0}$

void

$\eta[\mathcal{A}]$

location

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

composition

$\diamond \mathcal{A}$

somewhere modality

$\diamond \mathcal{A}$

sometime modality

$\mathcal{A} @ \eta$

location adjunct

$\mathcal{A} \triangleright \mathcal{B}$

composition adjunct

$\forall x. \mathcal{A}$

universal quantification over names

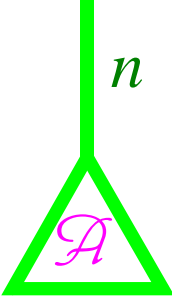
where η is a name n or a (quantifiable) variable x .

Basic Modalities


- $\mathbf{0}$: *here now* there is absolutely nothing ($n[\]$ abbreviates $n[\mathbf{0}]$):

$\mathbf{0}$ satisfied by *(void)* i.e. by $\mathbf{0}$

- $n[\mathcal{A}]$: *here now* there is exactly one edge called n , whose descendent satisfies (*there now*) the formula \mathcal{A} :

$n[\mathcal{A}]$ satisfied by  i.e. by $n[P]$
if $P \models \mathcal{A}$

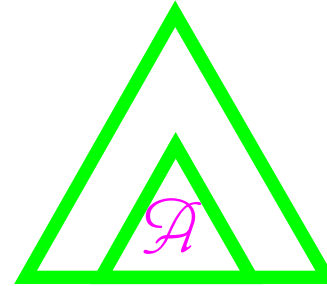
- $\mathcal{A} | \mathcal{B}$: *here now* there are exactly two things next to each other, one satisfying (*there now*) \mathcal{A} and one satisfying (*there now*) \mathcal{B} :

$\mathcal{A} | \mathcal{B}$ satisfied by  i.e. by $P | Q$
if $P \models \mathcal{A}$
and $Q \models \mathcal{B}$

-
- $\heartsuit A$: *somewhere now*, there is a place satisfying (*there now*) A :

$\heartsuit A$

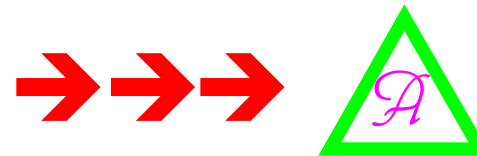
satisfied by



- $\diamond A$: *here sometime*, there is a thing satisfying (*here then*) A :

$\diamond A$

satisfied by



Derived Modalities

- *Everywhere* \mathcal{A} :

$$\Box \mathcal{A} \triangleq \neg \Diamond \neg \mathcal{A}$$

What is true *everywhere*? Not much, unless qualified:

$$\Box (\mathcal{A} \Rightarrow \mathcal{B})$$

everywhere \mathcal{A} is true, \mathcal{B} is true as well

- *Always* \mathcal{A} :

$$\Box \mathcal{A} \triangleq \neg \Diamond \neg \mathcal{A}$$

What will *always* be there? (Structure invariant:)

$$\Box \text{Pisa}[\text{LeaningTower}[\dots] \mid \dots]$$

Other Logical Connectives

- Anything (including void)

T (Anything satisfies it.) A.k.a.: ..

- Normal implication

$$\mathcal{A} \Rightarrow \mathcal{B}$$

if \mathcal{A} is true here now, then \mathcal{B} is true here now

$$\text{Borders}[..] \Rightarrow \text{Borders}[\text{Starbucks}[..] | ..]$$

If there is a Borders bookstore, there is a Starbucks inside.

$$(\text{NonSmoker}[..] | ..) \Rightarrow (\text{NonSmoker}[..] | \text{Smoker}[..] | ..)$$

If there is a non-smoker, there is nearby a smoker.

Spatial Implications

- Parallel implication

$$\mathcal{A} \mid \Rightarrow \mathcal{B} \triangleq \neg(\mathcal{A} \mid \neg \mathcal{B})$$

It is not possible to split the current location in such a way that one part satisfies \mathcal{A} and the other does not satisfy \mathcal{B} .

In other words, every way we split the current location, if one part satisfies \mathcal{A} , then the other part must satisfy \mathcal{B} .

$$\square \text{ Bath} [\spadesuit (\text{NonSmoker}[\dots] \mid \Rightarrow \text{Smoker}[\dots] \mid \dots)]$$

It is always the case that at the Bath, anywhere there is a non-smoker there is, nearby, a smoker.

-
- Nested implication

$$n[\Rightarrow \mathcal{A}] \triangleq \neg n[\neg \mathcal{A}]$$

It is not possible that the contents of an n location do not satisfy \mathcal{A} .

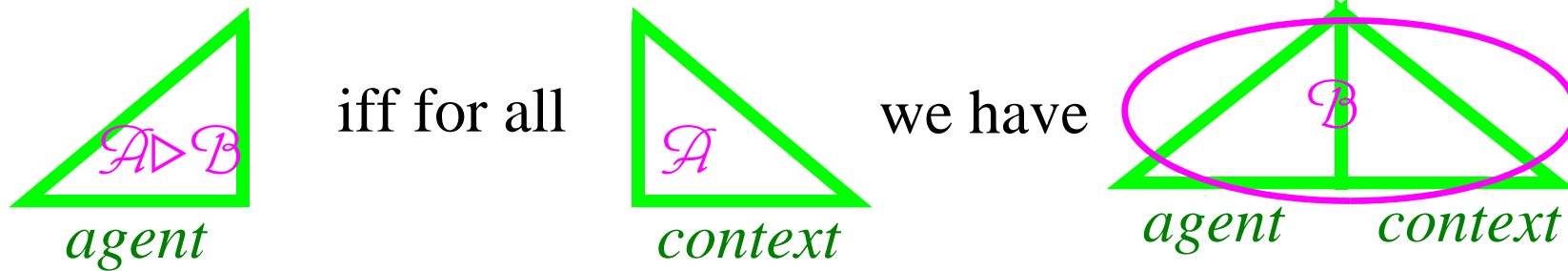
In other words, if there is an n location, its contents satisfy \mathcal{A} .

$$\square \text{US}[\forall \text{Borders}[\Rightarrow \text{Starbucks}[\dots] \mid \dots]]$$

Everywhere in the US, there is always a Starbucks inside a Borders.

Adjunctions

- $A \triangleright B$: even when the agent is in presence of any context (e.g.: "attacker") bound to satisfy A , the system satisfies B .



- Example (from two logically contradictory points of view):

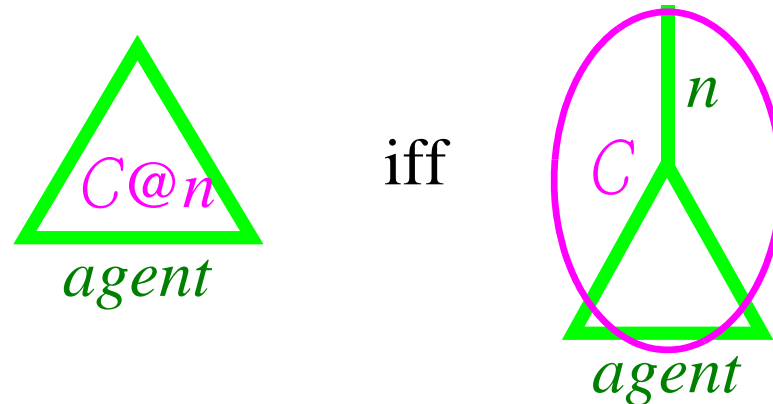
$$\text{bait}[\dots] \models \text{fish}[\dots] \triangleright \diamond \text{fish}[\text{bait}[\dots] \mid \dots]$$

$$\text{fish}[\dots] \models \text{bait}[\dots] \triangleright \square(\text{fish}[\dots] \mid \text{bait}[\dots])$$

Bait wants to catch fish. Fish wants to avoid bait.

- A logical adjunction: $(A \mid B) \Rightarrow C$ iff $A \Rightarrow (B \triangleright C)$

- $C@n$: even when the agent is ("thrown") in a location n , the system satisfies C .



- Example: one would hope that fish[...] satisfies:

$(\Box \text{tank}[\text{fish}[..] | ..]) @ \text{tank}$

A fish will survive in a tank.

- A logical adjunction: $n[\mathcal{A}] \Rightarrow C$ iff $\mathcal{A} \Rightarrow C@n$.

Matches and Queries for SSDB Applications

- Add matching variables to the logic (generalizing $..$) and enrich the satisfaction relation with a matching environment:
 - So that $P \models n[\mathcal{X} \mid m[\mathcal{Y}]]$ produces bindings for \mathcal{X} and \mathcal{Y} .
- Add a sophisticated sublanguage for matching paths, in addition to the existing (already quite rich) possibilities:
 - exact: $n[m[p[\mathcal{X}]] \mid ..]$
 - dislocated: $n[\diamond(m[\mathcal{X}] \mid ..)]$ an n , an arbitrary path, then an m
 - disjunctive: $n[p[\mathcal{X}]] \vee m[p[\mathcal{X}]]$
 - negative: $\diamond m[\neg(p[..] \mid ..) \mid q[\mathcal{X}]]$ in some m , in a q not next to a p
 - wildcard and restricted wildcard: $m[\exists x.x \neq n \wedge x[\mathcal{Y}]]$

Adjunctive Queries

- Using adjunctions, we can express some pretty fancy queries:
 - $\diamond m[\mathcal{X}@n]$:
take the data D under an m somewhere, and bind \mathcal{X} to $n[D]$.
(Because: $D \models \mathcal{X}@n$ iff $n[D] \models \mathcal{X}$, so bind \mathcal{X} to $n[D]$.)
 - $\diamond m[q[\mathcal{Y}] \mid (r[\mathcal{Y}] \triangleright \mathcal{X})]$:
take all the data D which is somewhere under an m and next to a $q[P]$ (with \mathcal{Y} bound to P), and bind \mathcal{X} to $D \mid r[P]$.
(Because: $D \models r[\mathcal{Y}] \triangleright \mathcal{X}$ iff $\forall Q \models r[\mathcal{Y}]. D \mid Q \models \mathcal{X}$. Since $r[P] \models r[\mathcal{Y}]$, take $Q = r[P]$ and bind \mathcal{X} to $D \mid r[P]$.)
(May want to mark the first \mathcal{Y} as the binding occurrence.)
- Not clear what how much expressive power we have here, but the idea of using adjunctions to express query-and-recombination situations seems interesting, and it came out of existing operators.

Thesis Instance: Query Optimization

Hmmm... The query language may work out ok. But what's a query language without optimizing transformations? Without a query algebra?

Wait... *algebra?*? We got algebra over logical operators!
More than that, we got tons of inference rules!

- Given the satisfaction relation, we can infer valid logical equivalences and implications. We have lots of those. We can conceivably use them as a guide to optimizing transformations. E.g.:

$$A \mid \mathbf{0} = A$$

$$A \mid B = B \mid A$$

Thesis Instance: Update

Hmmm... Bulk, DB-like, yet semistructured update. Yet another contradiction. No way it can ever work. But what's a DB without update? Without a way of changing the funny trees.

Wait... *changing the funny trees!!* That's all Ambients are good for. We got semistructured data, we got semistructured computation, hence we got semistructured update!

Further, in the logic we have ways of *specifying* updates. (Leaving the unenviable task of actually doing it to all that wonderful DB optimization technology.)

Back to Satisfaction

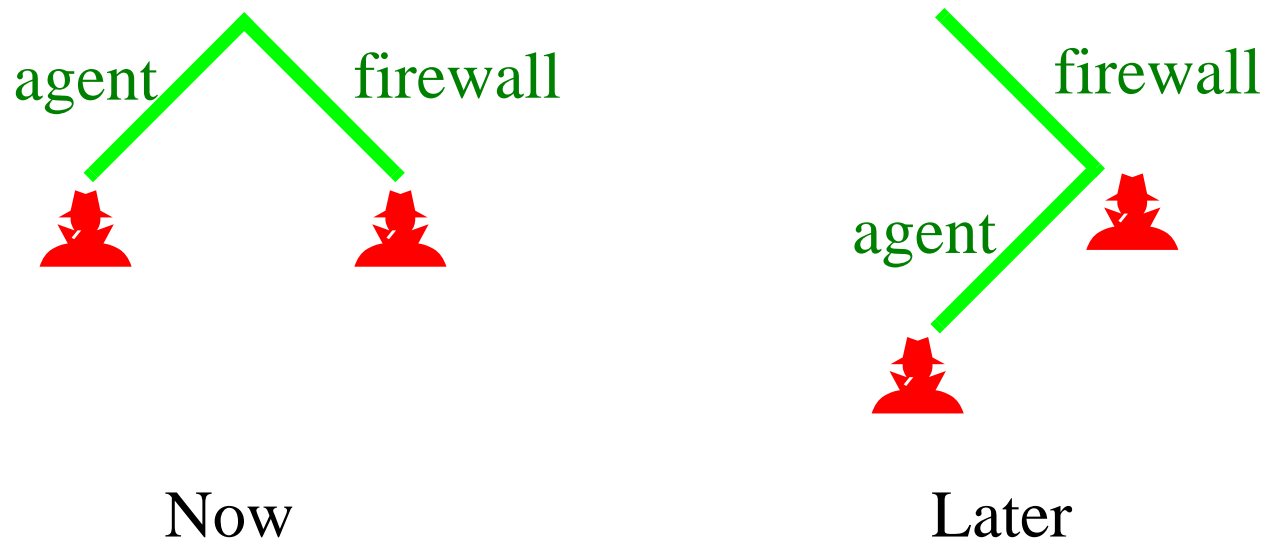
- There is a satisfaction relation between a tree and a formula.

$$P \models \mathcal{A}$$

- Temporally, this is (in finite cases) ordinary model-checking.
 - Temporally+spatially, this can be understood as a transformation process... that is:
- An update.
 - $P \models \mathcal{A} \triangleright \diamond \mathcal{B}$ means: the process P , in presence of any data matching \mathcal{A} eventually produces data matching \mathcal{B} . (Optionally, variables may relate parts \mathcal{A} of to parts of \mathcal{B} .)
 - Finding a process that satisfies $\mathcal{A} \triangleright \diamond \mathcal{B}$ means finding an update procedure.

Expressing Properties of Mobile Computation

- These properties 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.



- N.B. This could be the spec of a database update.

Conclusions

- Semistructured data and mobile computation are naturally related.
*P*roof: most of these slides were *not* prepared for this talk!
- So, we have an unexpected connection:
 - With slight modifications, our spatial logic can be seen as a *query-language* for semistructured data.
 - The ambient calculus can be seen as a *computational model* over semistructured data. (E.g. for database updates.)
 - Type systems for the ambient calculus can be seen as *weak schemas* for semistructured data.
- Much to be done yet, to flesh out this connection.