Logic and distributive laws

The twisted monads

Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC

Agenda

Terminology and level set
- Short break
- Definitions and examples
- Short break
- Hands-on calculation
- Short break
- Hands-on calculation
- Wrap up

Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC

Aims and measures of success

Aim: to give this idea away!

Measure of success:

- other people can make the calculations
- other people can write down, evaluate and extend the core algorithms
- other people understand the motivation for this framework of calculation

Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC

Aims and measures of success

What is presented is model-theoretic.

Extra credit: someone (besides me) presents the extension of the algorithm to the proof-theoretic

- i.e. they write down the operation of the algorithm on morphisms

Motivation

- (Behavioral and structural queries in physical systems) Find all the pathways, p, in <SignallingPathwayDB> such that when <Reagent> is added in concentration K p eventually reaches states in which we see these <Reagent1>,...,<ReagentN> in concentrations K1,...,KN.

- (Behavioral and structural queries in logical systems) Find all the implementations of message queue in http://svn.myrepo.org such that the queue is FIFO and use the Xerces XML parser for XML-encoded messages

- (Structural queries in physical systems) Find all the knots in <KnotDB> that have alternations of <Trefoil> and any 7-crossing knot as subknots

- (Geometric queries in physical systems) Find all the conduit wires running from the nose of the aircraft to the midsection that are likely to undergo a twist between source and destination

Terminology & level setting

Claim I : A logic is comprised of three data

A monad, `T`, for a term language (the witnesses of propositions)

A monad, `S`, for collecting witnesses

A distributive law l : `TS -> ST` illustrating how terms built over collections (formulae) are interpreted as collections of witnesses

Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC

Terminology & level setting

Part of our mission is to develop enough common understanding that this claim makes sense and can be independently verified

Terminology & level setting

Claim II : A model-checker is a database turned sideways, that is, given

a logical language, L, and

a model-checker for checking assertions of the form t |= l

a store of terms, src

we can describe an algorithmic approach to selecting all the terms in src satisfying l, that is interpreting statements of the form

SELECT t FROM src WHERE l

Terminology & level setting

Part of our mission can be nearly immediately discharged: there's an obvious brute-force algorithm that will do this. What is it?

Another part of our mission is to convince you that there are much more efficient algorithms that compile the model-check into a query plan over a programming language suitably extended with the query operations of the native query language of an industrial scale store.

That is, part of our mission is to convince you that there's an extension of LINQ that makes the observation useful for industrial scale applications.

Terminology & level setting

Claim I : A logic is comprised of three data

A monad, `T`, for a term language (the witnesses of propositions)

A monad, `S`, for collecting witnesses

A distributive law `l : TS -> ST` illustrating how terms built over collections (formulae) are interpreted as collections of witnesses

So, we have to know enough category theory and standard categorical interpretation of logic to unpack this statement.

Terminology & level setting

Show of hands, please:

You can come to the whiteboard, write down a definition of

a category

a functor

a natural transformation

a monad

a distributive law

and present it to us

Terminology & level setting

Show of hands, please:

- how a free algebra is represented by a monad
- how non-free versions are related to the Kleisli and/or Eilenberg-Moore algebra of a monad

extra credit

- how is this related to Universal Algebra and standard presentations of algebras via generators and relations

Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC

Terminology & level setting

Show of hands, please:

the standard categorical interpretation of logic

the propositions-as-types or Curry-Howard paradigm

Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC

Terminology & level setting

Along the way we will make use of a combination of tools

EBNF

Structural equivalence

Reduction relations

These are the modern computer scientist's way of writing down an algebra + a structure map together with a rewriting form of computation

Definitions & examples

Defn: A category is ...

Defn: A functor is ...

Defn: A natural transformation is ...

Defn: A monad is ...

Defn: An algebra for a monad is ...

Defn: A distributive law is ...

Definitions & examples

Just a thought: Could monads be the right computational unit for category theory?

Spse C a(n arrows-only presentation of a) category, then

|C| =

```
for( morph <- C ) yield { src( morph ) }
                  ++ for( morph <- C ) yield { trgt( morph ) }
```

Definitions & examples

First example: deriving separation logic from this approach

Defn: A monoid is

a set, M, together with

a distinguished element, e

a binary operation _ * _ : M x M -> M

satisfying

e * m = m = m * e

m1 * (m2 * m3) = (m1 * m2) * m2

Definitions & examples

An alternative presentation

Grammar

m,n ::= e

g1 | ... | gN

m*n

Structural equivalence (the smallest equivalence relation containing)

m * e = m = e * m

m1 * (m2 * m3) = (m1 * m2) * m3

Definitions & examples

```
k & l ::=
  true        [| true |](v) = L(m)
  ~k          [| ~k |](v) = L(m) \ [| k |]
  k & l       [| k & l |](v) = [| k |](v) \cap [| l |](v)

  e           [| e |](v) = { m \in L(m) | m = e }
  gi          [| gi |](v) = { m \in L(m) | m = gi }
  k * l       [| k * l |](v) =
                { m \in L(M) | m = m1 * m2, m1 \in [| k |](v), m2 \in [| l |](v) }

  rec X.k     [| rec X.k |](v) = U{ S \in \Pow(L(m)) | [| k |](S/X:v) = S }
  X           [| X |](v) = v(X)
```

Definitions & examples

A simple use

```
Prime = ~e & ~(~e * ~e)
```

Definitions & examples

What has this got to do with category theory?

Where are the monads?

Where are the algebras?

Where is the distributive law?

How does this relate to Universal Algebra?

How does this relate to Curry-Howard?

Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC

Definitions & examples

An alternative view of the lambda calculus

Grammar

```
m,n ::= 0 | @x | \x -> m | m x

x ::= «m»
```

Definitions & examples

An alternative view of the lambda calculus

Structural equivalence: alpha-equivalence + a little more

Reduction relation depends on a slightly different notion of substitution

Definitions & examples

```
k,l ::= true                 [| true |](v) = L(m)
        ~k                    [| ~k |](v) = L(m) \ [| k |]
        k & l                [| k & l |](v) = [| k |](v) \cap [| l |](v)

        \ a -> k          [| \ a -> k |](v) = { m \in L(m) | m = \ x -> m', x \in
[| a |](v), m' \in [| k |] }

        <k>l                 [| <k>l |](v) = { m \in L(m) | \exists n \in [| k |]. km ->
m', m' \in [| l |] }

        rec X.k          [| rec X.k |](v) = U{ S \in \Pow(L(m)) | [| k |](S/X:v) =
S }
        X                     [| X |](v) = v(X)

a,b ::= <<m>>        [| <<m>> |](v) = { <<n>> \in <<L(m)>> | m = n }
        <<k>>        [| <<k>> |](v) = { <<m>> \in <<L(m)>> | m \in [| k |] }
```

Definitions & examples

Where are the monads?

Where are the algebras?

Where is the distributive law?

How does this relate to Universal Algebra?

How does this relate to Curry-Howard?

Organizing principles

A domain (of discourse) is given by

A monad, T, for a term language (the witnesses of propositions)

A structural equivalence, ≡, on T, such that (T, ≡) is an algebra for T

A rewrite rule, ->, called the transition or reduction relation on T extending smoothly to T/≡.

Organizing principles

Examples:

A monoid is a domain in which the reduction relation is empty.

The lambda calculus is a domain in which the structural equivalence is alpha-equivalence and the reduction relation is beta-reduction.

The π-calculus is a domain in which the structural equivalence is normally called structural equivalence! and the rewrite rule is comm + the smooth extension to the algebra.

Organizing principles

In the physical sciences domains have historically been represented via some flavor of linear algebra. Note what this does to the packaging of dynamics.

Vector spaces just sit there. They enjoy no dynamics. You need transformations, such as linear transformations, to get them to dance.

Domains repackage dynamics with the algebraic structure. This does something crucial to the notion of morphism at the categorical level.

Organizing principles

The structure of vector spaces and linear transforms fits well with the classical development of category theory

The relationship of domains to category theory is a work in progress

- In historical terms, it’s brand new
- In categorical terms, we have to understand what is to be preserved when we preserve dynamics -- without this understanding there is no good definition of morphism

Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC

Organizing principles

Domains enjoy something that is a profound gap in classical models of dynamics: an effective notion of equivalence of dynamical systems

- Bisimulation is perhaps one of the most important ideas computer science will bring to the physical sciences

Calculations

Redo the monoid example, but this time instead of Set as the collection in which to accumulate witnesses, let's pick a different collection: sets of sequences of witnesses

If X is a set, Q[X] is the domain of all sets of sequences of elements of X, i.e.

$$Q[X] = Pow[X^*]$$

Hint: does anyone know what logic this corresponds to?

Calculations

Extra credit: how do we add probability into this setting?

Calculations

How do we compile structural queries into native storage query language?

How do we effect a query plan?

How does this relate to symbolic execution and behavior in time?

Motivation

- (Behavioral and structural queries in physical systems) Find all the pathways, p, in <SignallingPathwayDB> such that when <Reagent> is added in concentration K p eventually reaches states in which we see these <Reagent1>,...,<ReagentN> in concentrations K1,...,KN.
- Priami-Aviv encoding of cell signaling pathways identifies
  - suitable variant of π-calculus as the term language
  - suitable variant of Caires spatial-behavioral logic as the query language (algorithm agrees)

Motivation

- (Behavioral and structural queries in logical systems) Find all the implementations of message queue in http://svn.myrepo.org such that the queue is FIFO and use the Xerces XML parser for XML-encoded messages
- Berger-Honda-Yoshida encoding of core Java identifies
  - suitable variant of lambda calculus as term language
  - suitable variant of rely-guarantee logic as query language (algorithm agrees)

Motivation

- (Structural queries in physical systems) Find all the knots in <KnotDB> that have alternations of <Trefoil> and any 7-crossing knot as subknots
- Meredith-Snyder encoding of knots as process identifies
  - suitable variant of π-calculus as term language
  - suitable variant of Caires' spatial-behavioral logic as query language (algorithm agrees)

Motivation

- (Geometric queries in physical systems) Find all the conduit wires running from the nose of the aircraft to the midsection that are likely to undergo a twist between source and destination
  - Hestenes, et al identify Clifford algebra as term language
  - Use the algorithm to identify the logic

Where Biosimilarity is in the implementation of this idea

SpecialK/KVDB - an open source, distributed key-value database where the keys are prolog terms

Stratolytics - an analytics as a service platform

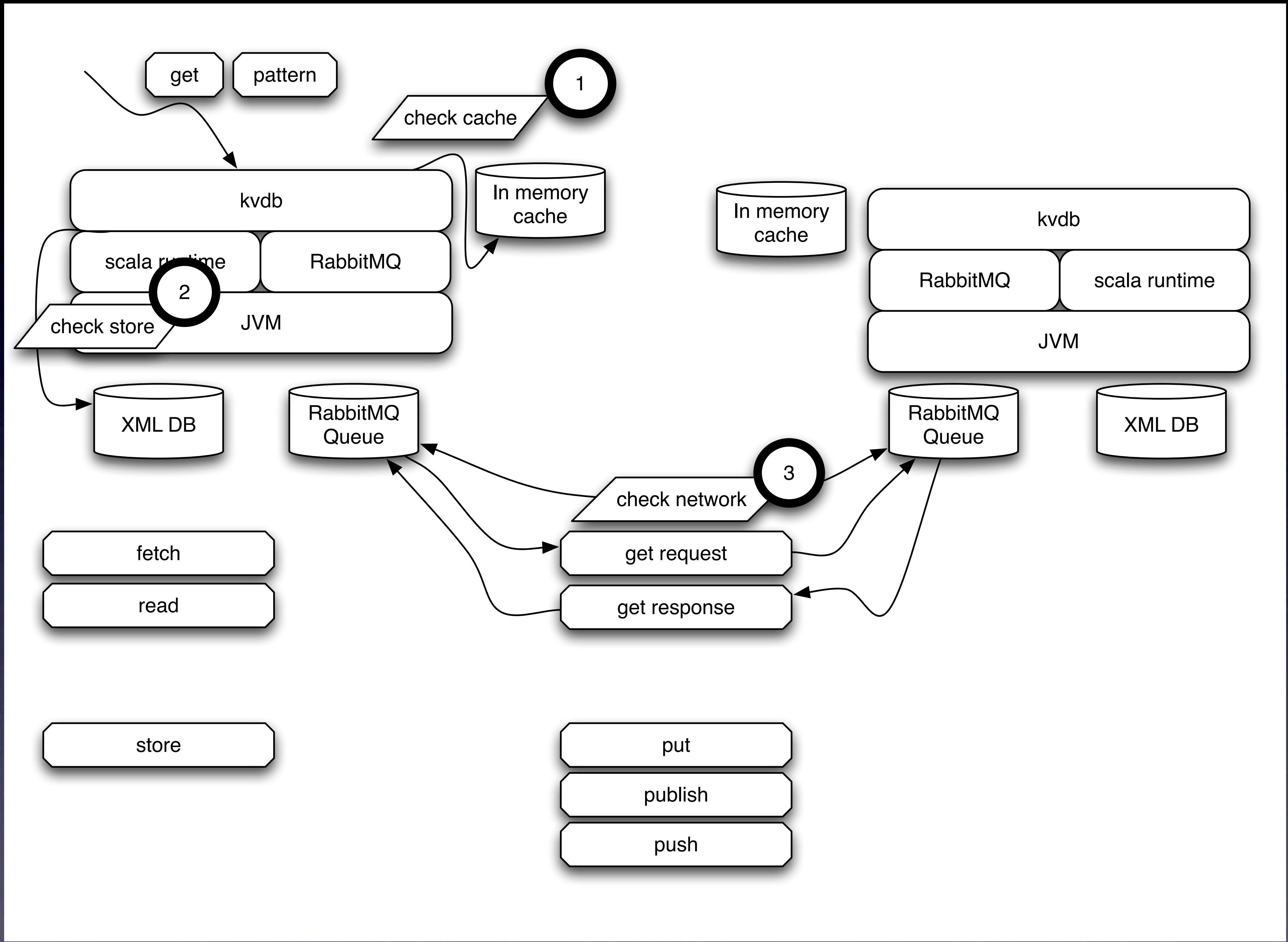
Quality ***o***f ***I***dentity - an identity management platform

Where Biosimilarity is in the implementation of this idea

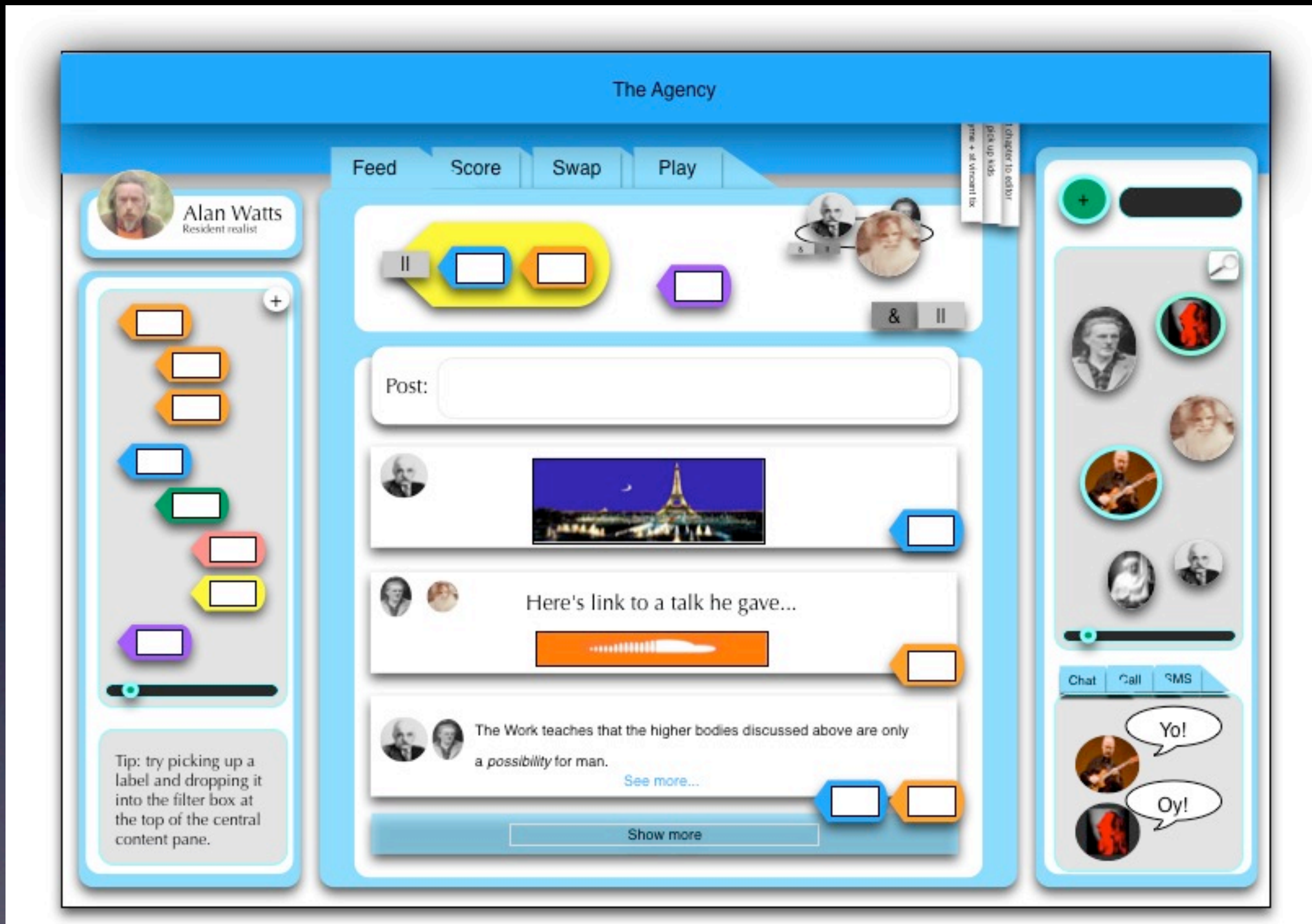
```
for( e <- kvdb.get( cnxn )( lbl ) { handle( e ) }
```

Analysts wire models directly to web-fed, label-filtered, cnxn-vetted sources to produce graphs, charts and decision support analysis.

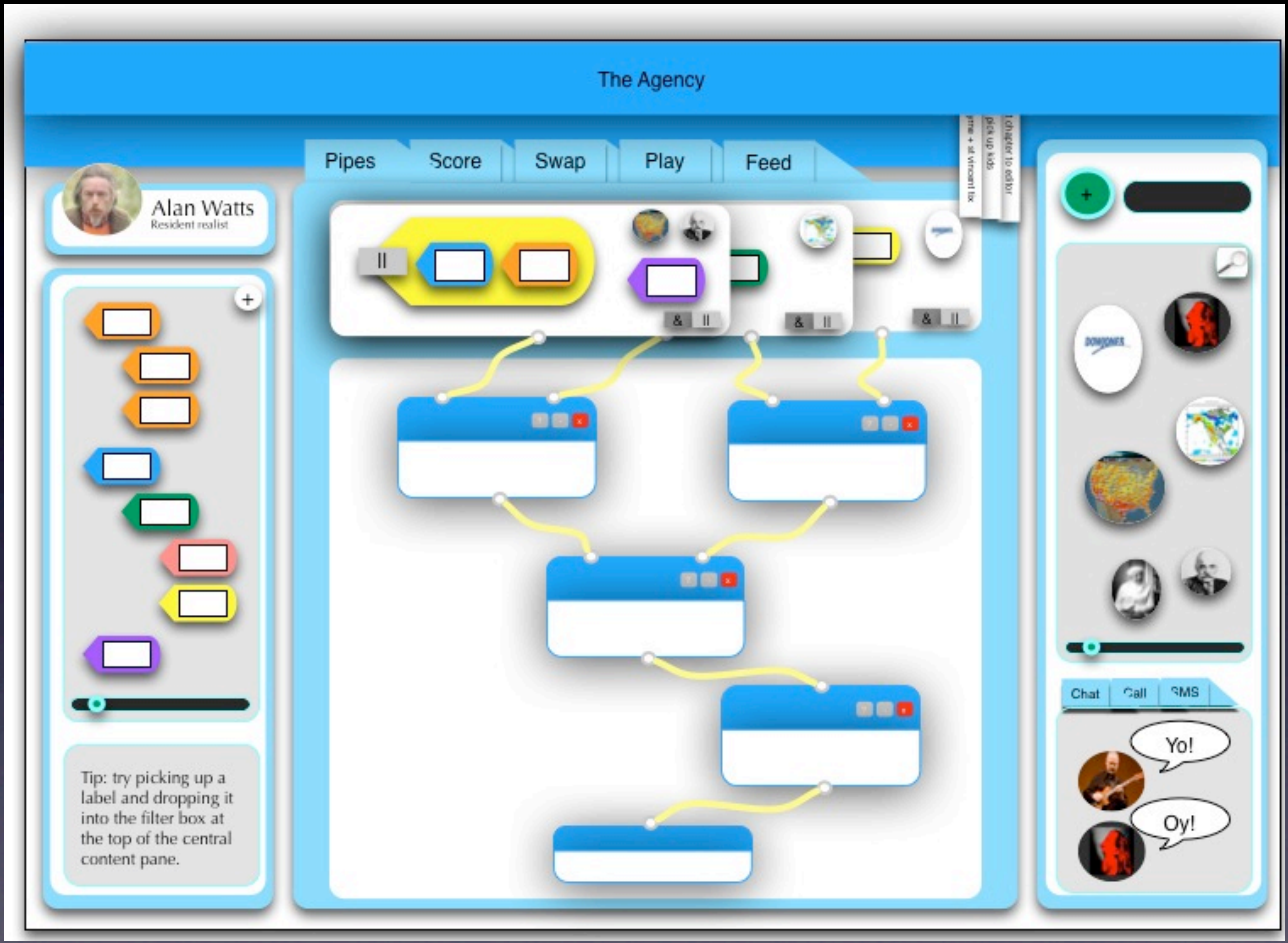
Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC



Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC



Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC



Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC

Conclusions

We have described an algorithm for generating a domain specific query engine

We have outlined motivations for addressing physical (signaling), logical and geometric systems in this way

We have outlined how such a system fits into a distributed cloud-based analytics-as-a-service offering supporting appropriate individual and organization privacy

Questions?

Lucius Gregory Meredith • Managing Partner • Biosimilarity, LLC

Backup

Definitions & examples

Defn: A category, *C*, is given in terms of the following pieces of data

|*C*| the collection of objects (aka types) of *C*

||*C*|| the collection of morphisms (aka programs) of *C*

src : ||C|| -> |C|, trgt : ||C|| -> |C|, id : |C| -> ||C||

_ • _ : ||C|| x ||C|| -> ||C|| + 1

Definitions & examples

satisfying:

m1 • m2 is defined when src(m2) = trgt(m1)

m • id(trgt(m)) = m = id(src(m)) • m

m1 • (m2 • m3) = (m1 • m2) • m3

Definitions & examples

Defn: A functor, F : C -> D, is a morphism in the category of categories, i.e. F decomposes into

|F| : |C| -> |D| the map on collection of objects of C

||F|| : ||C|| -> ||D|| the map on the collection of morphisms of C

preserving src, trgt, id and •

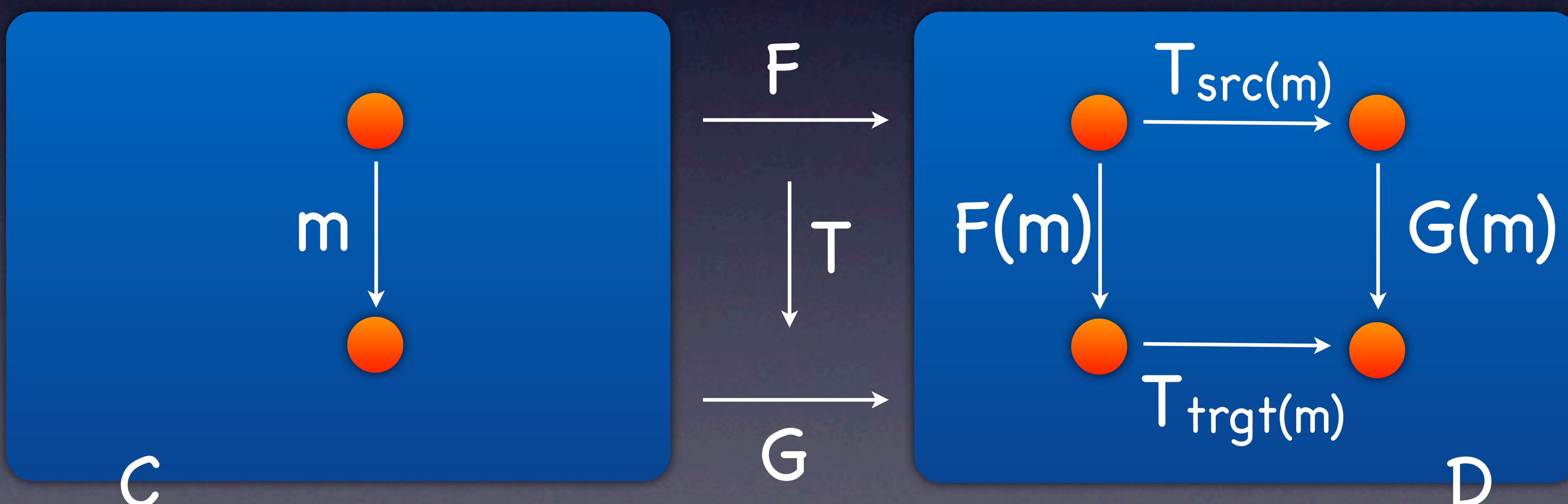
F(src(m)) = src(F(m)), F(trgt(m)) = trgt(F(m))

F(id(src(m))) = id(F(src(m)))

F(m1 • m2) = F(m1) • F(m2)

Definitions & examples

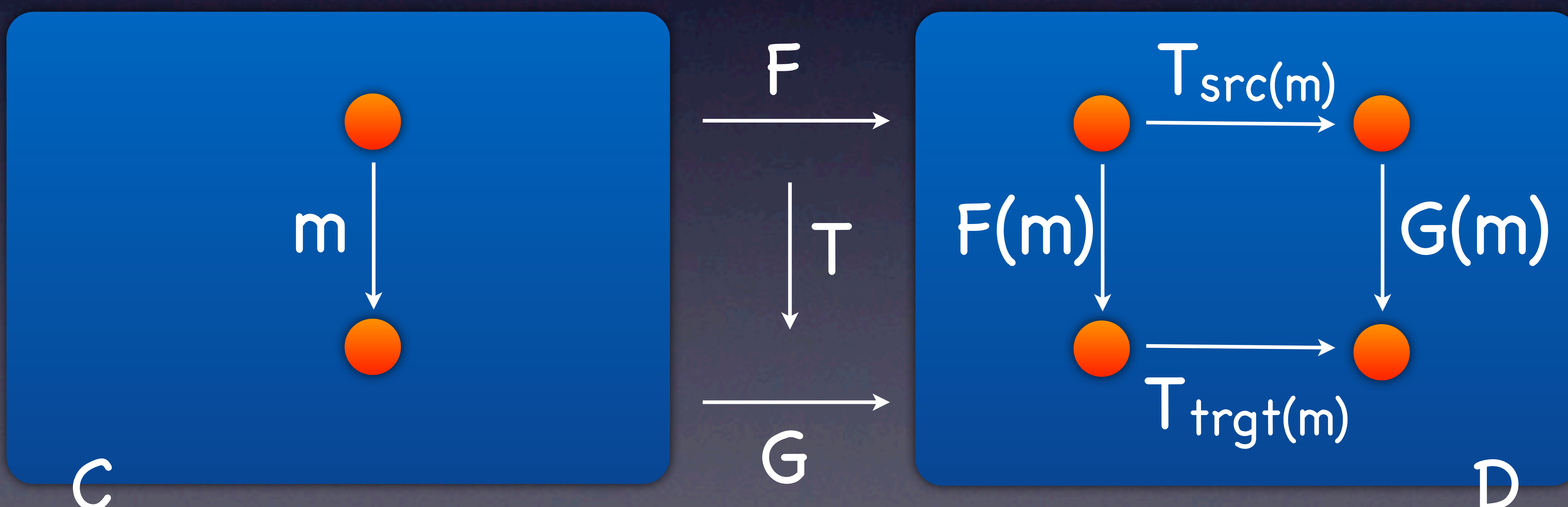
Defn: A natural transformation T : F -> G, is a morphism between functors, F, G : C -> D, i.e. it decomposes into a C-indexed family of morphisms, T_x, in D taking the image of F to the image of G



Definitions & examples

satisyfing:

$$F(m) \bullet T_{trgt(m)} = T_{src(m)} \bullet G(m)$$



Definitions & examples

Defn: A monad, T, is a monoid object in the category of categories, i.e. T decomposes into

T : C -> C an endo-functor on C

unit(T) : Id -> T a natural transformation from the identity functor, Id, to T

mult(T) : T^2 -> T a natural transformation

Definitions & examples

such that unit(T) and mult(T) interact coherently, i.e.

T mult • mult = mult • mult T

T unit • mult = unit T • mult

where

T mult is the evident natural transform from T^3 to T^2

mult T is the evident natural transform from T^3 to T^2

T unit is the evident natural transform from T^2 to T

unit T is the evident natural transform from T^2 to T

Definitions & examples

Defn: An algebra for a monad, T, is given by a pair (x, h) where

x in |src(T)|

src(h) = T(x), trgt(h) = x

$unit_x \bullet h = id_x$

$mult_x \bullet h = T(h) \bullet h$

Intuition : the pair (x, h) gives the model of the algebra. In the case of pure syntax -- meaning those x in |src(T)| built from term structure along -- this is tantamount to giving the structural equivalence.

Definitions & examples

Defn: A distributive law, l : ST -> TS, is a morphism between the composition of two monads, S, T with the units, mults and l all interacting coherently

T unit(S) • l = unit(S) T, S unit(T) • l = unit(T) S

S l • l S • T mult(S) = mult(S) T • l

l T • T l • mult(T) S = S mult(T) • l

Key point: The composition of monads is not necessarily a monad. With a distributive law, TS canonically acquires the structure of a monad.

Calculations

Doberkat shows how the algebras of the Giry monad on the category of Polish spaces (separable metric spaces with a complete metric) correspond to convex partitions of all probability measures.

What is not clear is how useful this result is in extending discrete computations to probabilistic ones.

Calculations

Our framework refines these questions:

- Structural equivalence can be probabilistic
  - We recognize the shape of this protein with confidence, x
- Transition rules can be probabilistic
  - These proteins are likely to undergo phosphorylation with probability, x