Sketch Theory as a Framework for Knowledge Management

NASA IV&V Workshop

9–11 September 2014

Morgantown, WV

Dr. Ralph L. Wojtowicz

Shepherd University
Shepherdstown, WV
rwojtowi@shepherd.edu

Baker Mountain Research Corporation
Yellow Spring, WV
ralphw@bakermountain.org

Science Technology Service

Representing Knowledge and Reasoning

Mathematical Logic

(Frege 1879)

$$\frac{A \Rightarrow B \qquad B \Rightarrow C}{A \Rightarrow C}$$

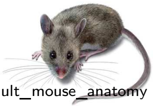
Databases + SQL

(Codd 1968)

part	name	price	quantity
2537	resistors	$ 5.99	10
2948	servo motors	$12.99	5
3647	Arduinos	$31.99	52

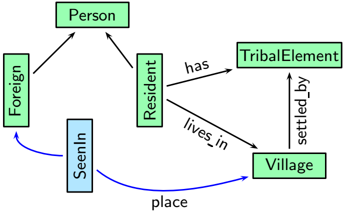
Semantic Web OWL/RDF + Description Logic (1999)

```
<rdfs:label xml:lang="en">vibrissa</rdfs:label>
<oboInOwl:hasRelatedSynonym>
  <oboInOwl:Synonym>
    <rdfs:label xml:lang="en">whisker</rdfs:label>
  </oboInOwl:Synonym>
</oboInOwl:hasRelatedSynonym>
```



www.berkeleybop.org/ontologies/owl/adult_mouse_anatomy

Sketches (Johnson-Rosebrugh 2000) + Q-Sequences (Freyd-Scedrov 1990)



Opportunity: Limitations of Knowledge Technologies

- Mathematical Logic
  - Computational complexity of some predicate calculus fragments
  - Complexity of the syntactic category used for knowledge alignment
  - Challenging to develop a human interface
- Databases + SQL
  - Limited notion of context/view (a single table)
  - Static schema
- Semantic Web OWL/RDF + Description Logic
  - Lack of modularity: meta-data, instance data and uncertainty integrated into a monolithic ontology
  - Limited compositional algebra: (disjoint) unions of ontologies
  - Need for constraint-preserving maps
- Sketch Theory
  - Meager computational infrastructure (e.g., relative to Jena)

Sketch Theory Strengths

Facts of Knowledge Models

Storage	Queries
Constraints	Uncertainty
Alignment	Dynamics
Context/Views	Software
Reasoning	Decision-Making
Translations	Human Interface

Knowledge Technologies

- Mathematical Logic (1879)
- Databases + SQL (1968)
- Semantic Web OWL/RDF
  + Description Logic (1999)
- Sketches (1968/2000)
  + Q-Sequences (1990)

Sketch Theory: Overview

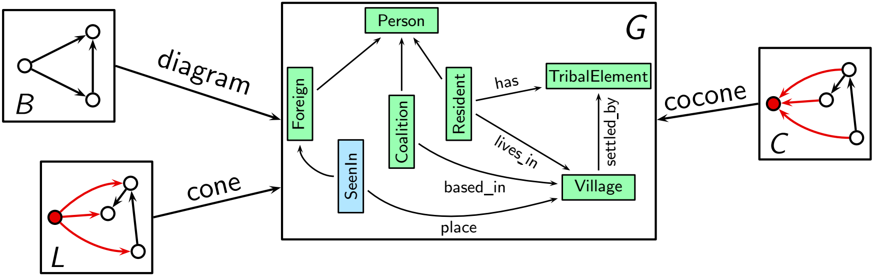
- Mature, graph-based foundation
- Vertices = classes or relations
- Edges = type information or maps
- Constraints/meta-data specified via graph maps (cones/cocones)
- Sketch maps respect constraints
- Grew from category theory in 1968
- Applied to data modeling since 1989

Sketch Theory: Strengths

- Visual/graphical modeling
- Modularity: data/concepts/uncertainty
- Combinatory algebra of sketches
- Concise graphical inference and inter-convertibility with 1st order logic
- Derived concepts via CW algorithm
- Rich composable views/context
- Dynamics via sketch maps

Sketch $(G, \mathcal{D}, \mathcal{L}, \mathcal{C})$

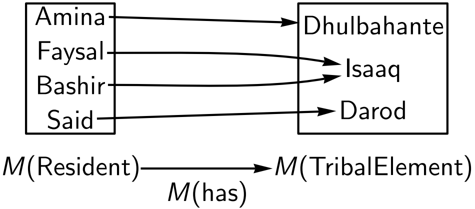
- A sketch $(G, \mathcal{D}, \mathcal{L}, \mathcal{C})$ consists of:
  - An underlying graph G
  - A set $\mathcal{D}$ of diagrams $B \to G$
  - A set $\mathcal{L}$ of cones $L \to G$
  - A set $\mathcal{C}$ of cocones $C \to G$
- The graph maps express the axioms or semantic constraints.



Set-Based Sketch Models

Set-based model of a graph

- Each vertex V is mapped to a set $M(V)$.
- Each edge $V \xrightarrow{e} W$ is mapped to a function $M(V) \xrightarrow{M(e)} M(W)$.

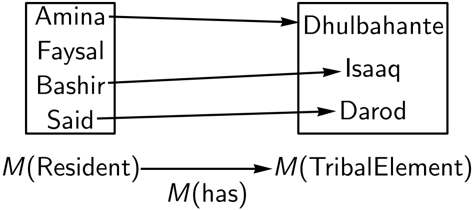


Set-based model of a sketch $(G, \mathcal{D}, \mathcal{L}, \mathcal{C})$

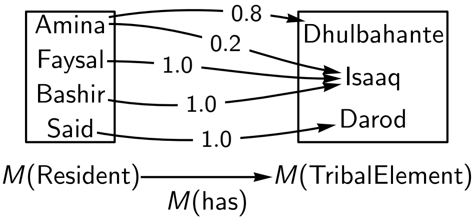
- A sketch model is, first, a model of the underlying graph G.
- Sketch constraints impose additional requirements on models.
- Expressiveness of the sketch imposes requirements on suitable categories of semantic models.

Sketch Models in Other Semantic Categories

- Partial function model of a graph edge Resident $\xrightarrow{\text{has}}$ TribalElement



- Stochastic matrix model of a graph edge Resident $\xrightarrow{\text{has}}$ TribalElement



Categorical Semantics of Sketches

- Vertices are interpreted as objects
- Edges are interpreted as morphisms
- Classes of constraints (cones and cocones) are distinguished by the shapes of their base graphs.
- Classes of sketches are distinguished by their classes of constraints.
- Like logics and OWL species, these have different expressive powers.

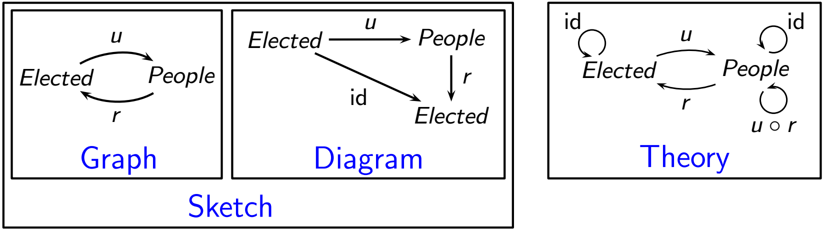
Small sample of the sketch semantics landscape

Sketch Class	Set	Partial Func.	Stoch. Matrices	Čencov Cat.	Prob. 0 Refl.	Dempster Shafer	Fuzzy Sets	Convex Sets
Regular	●	●	●	●	●	●	●	●
Finite Limit	●	●	×	×	×	×	●	●
Finite Coproduct	●	●	●	●	●	●	●	●
Entity-Attribute	●	●	×	×	×	×	●	●
Mixed	●	●	×	×	×	×	●	●

Civics Sketch $\mathbb{S}_1$

First formulation of civics concepts:

- Two classes: People and Elected officials
- People have Elected representatives via r.
- Elected officials are instances of people via u.
- Elected officials represent themselves via a diagram.



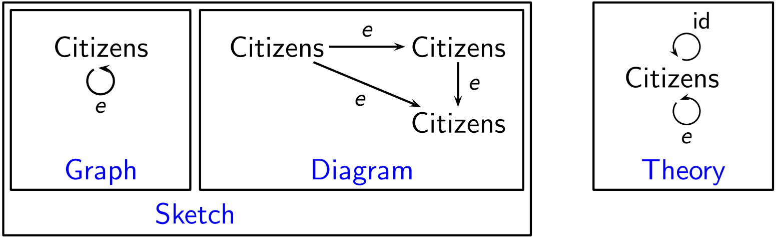
- The diagram truncates the infinite list of composites (property chains).

$$u \circ r \qquad r \circ u \qquad u \circ r \circ u \qquad r \circ u \circ r \qquad \cdots$$

Civics Sketch $\mathbb{S}_2$

Alternative formulation of the concepts:

- One class: Citizens
- Citizens have elected representatives via *e*.
- Elected officials represent themselves via a diagram.



- Number and names of vertices in $\mathbb{S}_1$ and $\mathbb{S}_2$ differ.
- The edges u and r of $\mathbb{S}_1$ have no corresponding edges in $\mathbb{S}_2$.
- The edge e of $\mathbb{S}_2$ has no corresponding edge in $\mathbb{S}_1$.

Civics Theory $\mathbb{T}_1$

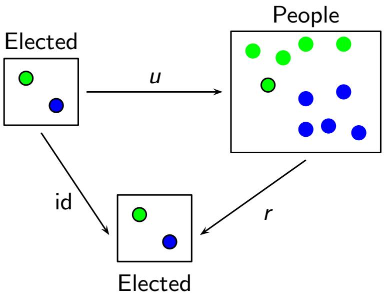
- **Sorts**: People, Elected
- **Function symbols**:

$$u : \text{Elected} \longrightarrow \text{People} \qquad r : \text{People} \longrightarrow \text{Elected}$$

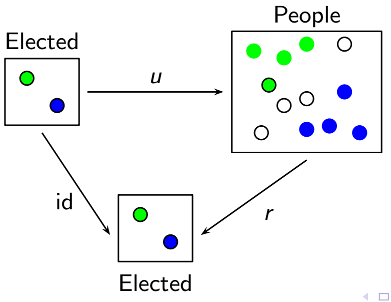
- **Axiom**: elected officials represent themselves

$$\top \vdash_x (r(u(x)) = x)$$

Set semantics



Partial function semantics



Civics Theory $\mathbb{T}_2$

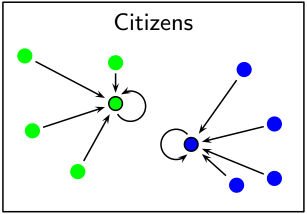
- **Sorts**: Citizens
- **Function symbols**:

$$e : \text{Citizens} \longrightarrow \text{Citizens}$$

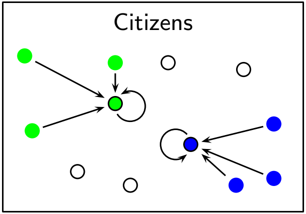
- **Axiom**: elected officials represent themselves

$$\top \vdash_x (e(e(x)) = e(x))$$

Set semantics



Partial function semantics



Logic-Based Inference: Sequent Calculus

Structural Rules[1]

$$(\varphi \vdash_{\vec{x}} \varphi) \qquad \frac{(\varphi \vdash_{\vec{x}} \psi)}{(\varphi[\vec{s}/\vec{x}] \vdash_{\vec{y}} \psi[\vec{s}/\vec{x}])} \qquad \frac{(\varphi \vdash_{\vec{x}} \psi) \; (\psi \vdash_{\vec{x}} \chi)}{(\varphi \vdash_{\vec{x}} \chi)}$$

Implication

$$\frac{((\varphi \wedge \psi) \vdash_{\vec{x}} \chi)}{(\varphi \vdash_{\vec{x}} (\psi \Rightarrow \chi))}$$

Equality

$$(\top \vdash_{x} (x = x))$$

$$((\vec{x} = \vec{y}) \wedge \varphi \vdash_{\vec{z}} \varphi[\vec{y}/\vec{x}])$$

Quantification[2]

$$\frac{(\varphi \vdash_{\vec{x},y} \psi)}{((\exists y)\varphi \vdash_{\vec{x}} \psi)} \qquad \frac{(\varphi \vdash_{\vec{x},y} \psi)}{(\varphi \vdash_{\vec{x}} (\forall y)\psi)}$$

Conjunction

$$(\varphi \vdash_{\vec{x}} \top) \qquad ((\varphi \wedge \psi) \vdash_{\vec{x}} \varphi) \qquad ((\varphi \wedge \psi) \vdash_{\vec{x}} \psi) \qquad \frac{(\varphi \vdash_{\vec{x}} \psi) \; (\varphi \vdash_{\vec{x}} \chi)}{(\varphi \vdash_{\vec{x}} (\psi \wedge \chi))}$$

Disjunction

$$(\bot \vdash_{\vec{x}} \varphi) \qquad (\varphi \vdash_{\vec{x}} (\varphi \vee \psi)) \qquad (\psi \vdash_{\vec{x}} (\varphi \vee \psi)) \qquad \frac{(\varphi \vdash_{\vec{x}} \chi) \; (\psi \vdash_{\vec{x}} \chi)}{((\varphi \vee \psi) \vdash_{\vec{x}} \chi)}$$

Distributive Axiom[3]

$$((\varphi \wedge (\psi \vee \chi) \vdash_{\vec{x}} (\varphi \wedge \psi) \vee (\varphi \wedge \chi))$$

Frobenius Axiom[3]

$$((\varphi \wedge ((\exists y)\psi) \vdash_{\vec{x}} (\exists y)(\varphi \wedge \psi))$$

Excluded Middle: $(\top \vdash_{\vec{x}} \varphi \vee \neg\varphi)$

Contexts are suitable for the formulae that occur on both sides of $\vdash$.

[1] In the substitution rule, $\vec{y}$ contains all the variables of $\vec{x}$.

[2] Bound variables do not also occur free in any sequent.

[3] These are required in coherent logic but are derivable in full first-order logic.

Proof of $(u(x) = u(y)) \vdash_{x,y} (x = y)$ for Civics Theory $\mathbb{T}_1$

1. $(u(x) = u(y)) \vdash_{x,y} (u(x) = u(y))$ . . . . Id
2. $(u(x) = u(y)) \vdash_{x,y} \top$ . . . . $\top$
3. $\top \vdash_{x} (r(u(x)) = x)$ . . . . axiom
4. $\top \vdash_{x,y} (r(u(x)) = x)$ . . . . Sub (3)
5. $\top \vdash_{x,y} (r(u(y)) = y)$ . . . . Sub (3)
6. $(x = y) \wedge (r(x) = z) \vdash_{x,y,z} (r(y) = z)$ Eq1
7. $(u(x) = u(y)) \wedge (r(u(x)) = x) \vdash_{x,y,z} (r(u(y)) = x)$ . . . . Subs (6)
8. $(u(x) = u(y)) \wedge (r(u(x)) = x) \vdash_{x,y} (r(u(y)) = x)$ . . . . Subs (7)
9. $(x = y) \vdash_{x,y} (y = x)$ . . . . previous proof
10. $(r(u(y)) = x) \vdash_{x,y} (x = r(u(y)))$ . . . . Subs (9)
11. $(u(x) = u(y)) \wedge (r(u(x)) = x) \vdash_{x,y} (x = r(u(y)))$ . . . . Cut (8), (10)
12. $(x = y) \wedge (y = z) \vdash_{x,y,z} (x = z)$ . . . . previous proof
13. $(x = r(u(y))) \wedge (r(u(y)) = y) \vdash_{x,y,z} (x = y)$ . . . . Subs (12)
14. $(x = r(u(y))) \wedge (r(u(y)) = y) \vdash_{x,y} (x = y)$ . . . . Subs (13)
15. $(u(x) = u(y)) \vdash_{x,y} (r(u(x)) = x)$ . . . . Cut (2), (4)
16. $(u(x) = u(y)) \vdash_{x,y} (u(x) = u(y)) \wedge (r(u(x)) = x)$ . . . . $\wedge$I (1), (15)
17. $(u(x) = u(y)) \vdash_{x,y} (x = (r(u(y)))$ . . . . Cut (16), (11)
18. $(u(x) = u(y)) \vdash_{x,y} (r(u(y)) = y)$ . . . . Cut (2), (5)
19. $(u(x) = u(y)) \vdash_{x,y} (x = r(u(y))) \wedge (r(u(y)) = y)$ . . . . $\wedge$I (17), (18)
20. $(u(x) = u(y)) \vdash_{x,y} (x = y)$ . . . . Cut (19), (14)

Prover9 Proof

- Input file:

```
formulas(assumptions).
  all x (r(u(x)) = x).
end_of_list.
formulas(goals).
  all x all y (u(x) = u(y)) -> (x = y).
end_of_list.
```

- Proof:

```
1 (all x r(u(x)) = x) ....................# label(non_clause).  [assumption].
2 (all x all y u(x) = u(y)) -> x = y ................# label(non_clause)
                                                      # label(goal).  [goal].
3 r(u(x)) = x.  ...................................[clausify(1)].
4 u(x) = u(y).  .......................................[deny(2)].
5 c2 != c1.  ..........................................[deny(2)].
6 x = y.  ..........................[para(4(a,1),3(a,1,1)),rewrite([3(2)])].
7 $F. ..................................................[resolve(6,a,5,a)].
```

- The shorter proof by contradiction uses classical first-order logic.
- First-order horn logic has lower computational complexity in general.

Sketch-Based Inference: Q-Sequences

- Idea: leverage the notion of Q-sequence to implement a reasoning engine for sketch-based knowledge models.
- P. J. Freyd and A. Scedrov. *Categories, Allegories*. 1990
- D. E. Rydeheard and R. M. Burstall. *Computational Category Theory*. 1988
- Analogy:

Q-sequence proof	$\Longleftrightarrow$	logical inference
functional programming (Haskell)	$\Longleftrightarrow$	procedural programming (C)

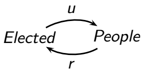
- A Q-sequence is a finite list of finitely-presented categories, maps and quantifiers Q_i.

$$A_0 \xrightarrow{Q_0} A_1 \xrightarrow{Q_1} A_2 \xrightarrow{Q_2} \cdots \xrightarrow{Q_{n-2}} A_{n-1} \xrightarrow{Q_{n-1}} A_n$$

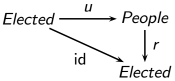
- Satisfaction of a Q-sequence in a category (sketch) is defined via a universal mapping property.

Q-Sequence Proof of a Subtype Property in $\mathbb{S}_1$

In civics sketch $\mathbb{S}_1$, we may conclude that Elected is a subclass of People.

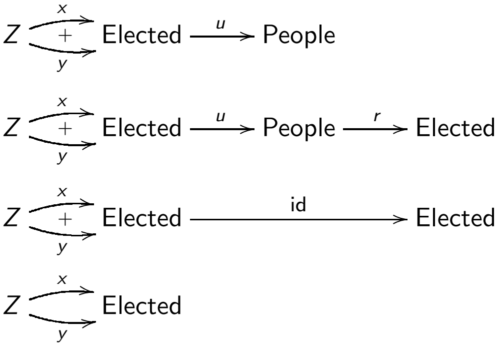


Graph



Diagram

Given any x and y as shown:



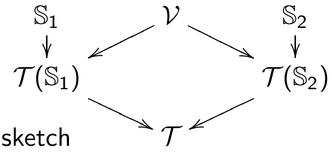
It follows that u is a monomorphism (one-to-one) in any model.

Presentations

- A sketch | first-order theory | ontology is a presentation of knowledge.
- Presentations generate additional knowledge needed for alignment.

Logical theory $\mathbb{T}$	syntactic category $\mathcal{C}_{\mathbb{T}}$
Ontology	rules
Sketch $\mathbb{S}$	theory of a sketch $\mathcal{T}(\mathbb{S})$

- Different presentations may generate equivalent structures.
- Sketches $\mathbb{S}_1$ and $\mathbb{S}_2$ representing common concepts are aligned by finding a sketch $\mathcal{V}$ and sketch maps as shown.



- Theory of a (linear) sketch
  - Carmody-Walters algorithm for computing left Kan extensions: generalizes Todd-Coxeter procedure used in computational group theory
  - Complexity difficult to characterize: can depend on order of constraints

Logic-Based Alignment: Provable Equivalence

- How can we align the civics theories?

$\mathbb{T}_1$	$\mathbb{T}_2$
u : Elected $\rightarrow$ People r : People $\rightarrow$ Elected $\top \vdash_x (r(u(x)) = x)$	e : Citizens $\rightarrow$ Citizens $\top \vdash_x (e(e(x)) = e(x))$

Logic-Based Alignment: Provable Equivalence

- How can we align the civics theories?

$\mathbb{T}_1$	$\mathbb{T}_2$
u : Elected $\to$ People r : People $\to$ Elected $\top \vdash_x (r(u(x)) = x)$	e : Citizens $\to$ Citizens $\top \vdash_x (e(e(x)) = e(x))$

- Provable equivalence: Every axiom of $\mathbb{T}_1$ is a theorem of $\mathbb{T}_2$ and conversely.

Logic-Based Alignment: Provable Equivalence

- How can we align the civics theories?

$\mathbb{T}_1$	$\mathbb{T}_2$
u : Elected $\to$ People r : People $\to$ Elected $\top \vdash_x (r(u(x)) = x)$	e : Citizens $\to$ Citizens $\top \vdash_x (e(e(x)) = e(x))$

- Provable equivalence: Every axiom of $\mathbb{T}_1$ is a theorem of $\mathbb{T}_2$ and conversely.
- This notion aligns theories that have the same signature.

Logic-Based Alignment: Provable Equivalence

- How can we align the civics theories?

$\mathbb{T}_1$	$\mathbb{T}_2$
u : Elected $\to$ People r : People $\to$ Elected $\top \vdash_x (r(u(x)) = x)$	e : Citizens $\to$ Citizens $\top \vdash_x (e(e(x)) = e(x))$

- Provable equivalence: Every axiom of $\mathbb{T}_1$ is a theorem of $\mathbb{T}_2$ and conversely.
- This notion aligns theories that have the same signature.
- Alignment typically involves use of derived concepts.

Logic-Based Alignment: Provable Equivalence

- How can we align the civics theories?

$\mathbb{T}_1$	$\mathbb{T}_2$
u : Elected $\rightarrow$ People r : People $\rightarrow$ Elected $\top \vdash_x (r(u(x)) = x)$	e : Citizens $\rightarrow$ Citizens $\top \vdash_x (e(e(x)) = e(x))$

- Provable equivalence: Every axiom of $\mathbb{T}_1$ is a theorem of $\mathbb{T}_2$ and conversely.
- This notion aligns theories that have the same signature.
- Alignment typically involves use of derived concepts.
- We need a concept that is less restrictive than provable equivalence.

Logic-Based Alignment: Morita Equivalence

- Theories $\mathbb{T}_1$ and $\mathbb{T}_2$ are Morita equivalent if their categories of models $\mathrm{Mod}_{\mathbb{T}}(\mathcal{D})$ (in any appropriate semantic category $\mathcal{D}$) are equivalent.

$$\mathrm{Mod}_{\mathbb{T}_1}(\mathcal{D}) \approx \mathrm{Mod}_{\mathbb{T}_2}(\mathcal{D})$$

Logic-Based Alignment: Morita Equivalence

- Theories $\mathbb{T}_1$ and $\mathbb{T}_2$ are Morita equivalent if their categories of models $\mathrm{Mod}_{\mathbb{T}}(\mathcal{D})$ (in any appropriate semantic category $\mathcal{D}$) are equivalent.

$$\mathrm{Mod}_{\mathbb{T}_1}(\mathcal{D}) \approx \mathrm{Mod}_{\mathbb{T}_2}(\mathcal{D})$$

- Theories are Morita equivalent iff their syntactic categories are.

$$\mathcal{C}_{\mathbb{T}_1} \approx \mathcal{C}_{\mathbb{T}_2}$$

Logic-Based Alignment: Morita Equivalence

- Theories $\mathbb{T}_1$ and $\mathbb{T}_2$ are Morita equivalent if their categories of models $\mathsf{Mod}_{\mathbb{T}}(\mathcal{D})$ (in any appropriate semantic category $\mathcal{D}$) are equivalent.

$$\mathsf{Mod}_{\mathbb{T}_1}(\mathcal{D}) \approx \mathsf{Mod}_{\mathbb{T}_2}(\mathcal{D})$$

- Theories are Morita equivalent iff their syntactic categories are.

$$\mathcal{C}_{\mathbb{T}_1} \approx \mathcal{C}_{\mathbb{T}_2}$$

- This notion solves the alignment problem for our civics theories.

Logic-Based Alignment: Morita Equivalence

- Theories $\mathbb{T}_1$ and $\mathbb{T}_2$ are Morita equivalent if their categories of models $\mathrm{Mod}_{\mathbb{T}}(\mathcal{D})$ (in any appropriate semantic category $\mathcal{D}$) are equivalent.

$$\mathrm{Mod}_{\mathbb{T}_1}(\mathcal{D}) \approx \mathrm{Mod}_{\mathbb{T}_2}(\mathcal{D})$$

- Theories are Morita equivalent iff their syntactic categories are.

$$\mathcal{C}_{\mathbb{T}_1} \approx \mathcal{C}_{\mathbb{T}_2}$$

- This notion solves the alignment problem for our civics theories.
- It can be difficult to use in practice: syntactic categories are infinite even for very simple theories.

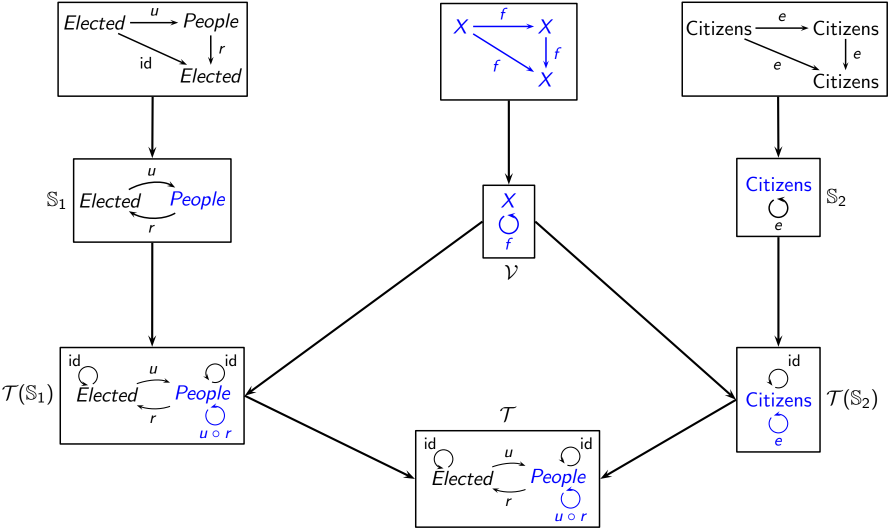
Logic-Based Alignment: Syntactic Categories

- The syntactic category $\mathcal{C}_{\mathbb{T}}$ of a theory $\mathbb{T}$ is constructed as follows:

objects:	α-equivalence classes of formulae-in-context: $\{\vec{x}.\varphi\}$
morphisms :	$\{\vec{x}.\varphi\} \xrightarrow{[\theta]} \{\vec{y}.\psi\}$ $\theta \vdash_{\vec{x},\vec{y}} \varphi \wedge \psi \qquad \varphi \vdash_{\vec{x}} (\exists \vec{y})\theta \qquad \theta \wedge \theta[\vec{z}/\vec{y}] \vdash_{\vec{x},\vec{y},\vec{z}} (\vec{z} = \vec{y})$
composition:	$\{\vec{x}.\varphi\} \xrightarrow{[\theta]} \{\vec{y}.\psi\}$, $\{\vec{y}.\psi\} \xrightarrow{[\gamma]} \{\vec{z}.\chi\}$, $\{\vec{x}.\varphi\} \xrightarrow{[(\exists \vec{y})(\theta \wedge \gamma)]} \{\vec{z}.\chi\}$
identity:	$\{\vec{x}.\varphi\} \xrightarrow{[\varphi \wedge (\vec{x}' = \vec{x})]} \{\vec{x'}.\varphi[\vec{x'}/\vec{x}]\}$

- We restrict the formulae φ and θ to be of the appropriate class: cartesian/regular/coherent/first-order.

Alignment of the Civics Sketches



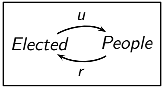
Sketch Views

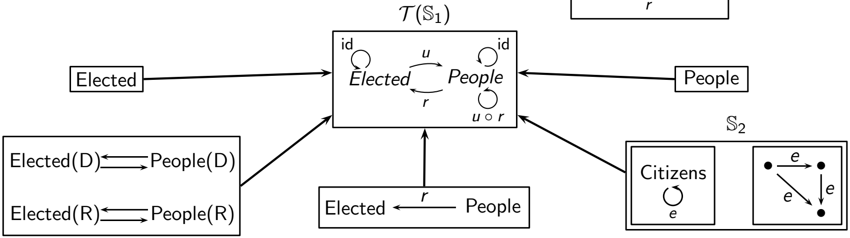
- A view $\mathcal{V} \Longrightarrow \mathbb{S}$ of a sketch $\mathbb{S}$ is a sketch $\mathcal{V}$ and a sketch map

$$\mathcal{V} \longrightarrow \mathcal{T}(\mathbb{S})$$

  from $\mathbb{S}$ to the theory $\mathcal{T}(\mathbb{S})$ of $\mathbb{S}$.

- Below are several views of the civics sketch $\mathbb{S}_1$ =





- A model of $\mathbb{S}$ induces a model of $\mathcal{T}(\mathbb{S})$ and of its views $\mathcal{V} \to \mathcal{T}(\mathbb{S}) \xrightarrow{M}$ Set.
- Views may be composed $\mathcal{V}_2 \Longrightarrow \mathcal{V}_1 \Longrightarrow \mathbb{S}$.
- View Update Problem: Under what conditions can updates to a model of $\mathcal{V}$ be propagated to a model of $\mathbb{S}$?

Challenge: Construct Views Tailored to Contexts

- Research area with narrower scope: context-sensitive Internet search
  - Google patent for "methods, systems and apparatus including computer program products, in which context can be used to rank search results" (USPTO 8,209,331 — 2012)
  - Yandex personalized web search challenge: www.kaggle.com
- Techniques to infer context from activities and rank data elements
  - Variable-length hidden Markov model
  - Parametric models of users
  - RankNet, LambdaRank, RankSVM
- Performance metrics used for context-sensitive rankings
  - Normalized discounted cumulative gain (scoring in Kaggle competition)
  - Kendall's τ comparison of rankings
  - Jaccard distance between top N rankings and target

Sketch Maps and Model Maps

- A sketch map $\mathbb{S}_1 \to \mathbb{S}_2$ is a graph map
$$G_1 \longrightarrow G_2$$
that preserves all the constraints of $\mathbb{S}_1$.
$$B \longrightarrow G_1 \longrightarrow G_2$$
- We use sketch maps to formulate the alignment problem.

- Given models M_1 and M_2 of a sketch $\mathbb{S}$, a model map $M_1 \to M_2$ is a collection of morphisms (one for each vertex V of G)
$$M_1(V) \xrightarrow{\tau_V} M_2(V)$$
that are consistent with the edges of G.
- Example:

$$\begin{array}{c} \text{Resident} \\ \Big\downarrow{\scriptstyle \text{live\_in}} \\ \text{Village} \end{array} \qquad \begin{array}{ccc} M_1(\text{Resident}) & \xrightarrow{\tau} & M_2(\text{Resident}) \\ {\scriptstyle M_1(\text{lives\_in})}\Big\downarrow & & \Big\downarrow{\scriptstyle M_2(\text{lives\_in})} \\ M_1(\text{Village}) & \xrightarrow[\tau]{} & M_2(\text{Village}) \end{array}$$

Transforming Sketches into Logical Theories

- Sketches are related to first-order logical theories by theorems of the form: Given any sketch $\mathbb{S}$ of class X, there is a logical theory $\mathbb{T}$ of class Y for which $\mathbb{S}$ and $\mathbb{T}$ have equivalent classes of models.
- D2.2 of Johnstone's *Sketches of an Elephant: A Topos Theory Compendium* gives explicit constructions of $\mathbb{T}$ from $\mathbb{S}$ and conversely.

Class of Sketches	Fragment of Predicate Calculus	Logical Connectives
finite limit	cartesian	$=, \top, \wedge, \exists^*$
regular	regular	$=, \top, \wedge, \exists$
coherent	coherent	$=, \top, \wedge, \exists, \bot, \vee$
geometric	geometric	$=, \top, \wedge, \exists, \bot, \bigvee$
σ-coherent	σ-coherent	$=, \top, \wedge, \exists, \bot, \bigvee_{i=1}^{\infty}$
finitary	σ-coherent	

* In cartesian logic, only certain existentially quantified formulae are allowed.

Software Infrastructure

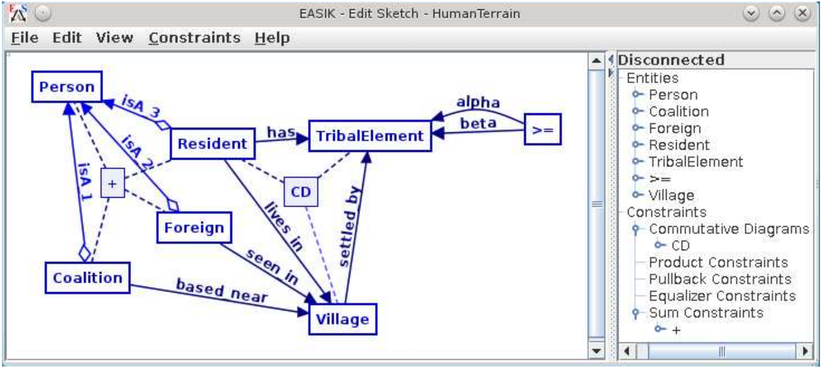
- Set-based models of entity-attribute sketches can be implemented using database features.

Sketch	database schema
Vertex	table with automatically-generated (`Serial`) key
Edge $A \xrightarrow{e} B$	foreign key in A-table referencing B-table key
Constraints	triggers

- Challenge: manage distributed sketch models, views and constraints — Google Megastore, Tenzing and Spanner; Apache Cassandra and Accumulo
- Reasoning
  - Transform to first-order theory then employ theorem prover
  - Q-sequence reasoning using computation category theory tools: Rydeheard and Burstall (ML implementations) 1988
- Theory of a (linear) sketch
  - Carmody-Walters algorithm for computing left Kan extensions: generalizes Todd-Coxeter procedure used in computational group theory
  - Complexity difficult to characterize: can depend on order of constraints

Easik Tool for Modeling with Sketches

- Entity Attribute Sketch Implementation Kit (Easik)
- http://mathcs.mta.ca/research/rosebrugh/Easik
- Build sketches, views and constraints
- Interface with MySQL or Postgres for (set-based models)
- No reasoning engine



Program

- The sketch data model demonstrates valuable features
  - Functional paradigm: syntax, models and maps
  - Separation of meta-data from instance data
  - Uncertainty and lack of information accounted for in models
  - Context-sensitive views which can be composed and combined
  - Formulation of the alignment problem using a well-defined mathematical construction (theory of a sketch)
  - Reasoning via graphical Q-sequences or transformation to predicate calculus fragment
- Research challenges
  - Implement sketch constraints on large, distributed models
  - Leverage insights, datasets and performance metrics from the narrower problem of context-sensitive Internet search
  - Develop and implement semi-automated alignment tool
  - Integrate reasoning and modeling algorithms with instance data into a common software platform
  - Characterize sketch classes corresponding to OWL species

References

- M. Barr and C. Wells. *Toposes Triples and Theories*. Springer-Verlag. 1985
- M. Barr and C. Wells. *Category Theory for Computing Sciences*. Prentice-Hall. 1990
- S. Carmody, M. Leeming and R. F. C. Walters. The Todd-Coxeter Procedure and Left Kan Extensions. *J. Symbolic Computation*. **19**:459–488. 1995
- P. Freyd and A. Scedrov. *Categories Allegories*. North-Holland. 1990
- J. W. Gray. The Category of Sketches as a Model for Algebraic Semantics. In *Categories in Computer Science and Logic*. V. 92 of Contemporary Mathematics. AMS. 1989
- M. Johnson and R. Rosebrugh. Sketch Data Models, Relational Schema and Data Specifications. *Electronic Notes in Theoretical Computer Science*. **61**(6):1–13. 2002
- M. Johnson, R. Rosebrugh and R. J. Wood. Lenses, Fibrations and Universal Translations. *Mathematical Structures in Computer Science*. **22**:25–42. 2012
- P. E. Johnstone. *Sketches of an Elephant: A Topos Theory Compendium*. Oxford University Press. 2002
- F. W. Lawvere and S. Schanuel. *Conceptual Mathematics*. Cambridge University Press. 2^{nd} Ed. 2009
- S. Mac Lane. *Categories for the Working Mathematician*. 2^{nd} Ed. Springer-Verlag. 1999
- D. E. Rydeheard and R. M. Burstall. *Computational Category Theory*. Prentice-Hall. 1988