Linguistics and Formal Ontology

Nicholas Asher

CNRS, Institut de Recherche en Informatique de Toulouse
Université Paul Sabatier

Vitoria, September 2014

Lexical semantics today

- formal lexical semantics
- distributional lexical semantics (unsupervised learning methods for building large lexicons completely automatically)

Why lexical semantics is important

- The meaning of a sentence, text or discourse is the result of a method of semantic composition and an assignment of meanings to basic terms.
- lexical semantics gives you the basic building blocks of meaning.
- good lexical semantics crucial not only for compositional semantics of the sentence but also for the interpretation of a whole text or discourse.

The State of Lexical Semantics

- lots of intriguing observation—polysemy, selectional restrictions, coercion, copredication, diathesis alternation
- lexical semantics' theoretical framework still unclear, but some new efforts (type theory, TCL)
  - Quine's ghost.
  - technical difficulties in extending Montague's framework to account for complex and systematic inferences involving word choice, semantic well-formedness.

Some paradigms

- ”cognitive semantics”: Talmy, Givon, etc.; some formalizations using algebraic methods (Gardenfors), little to no account of how meanings combine or of truth conditions.
- ”denotational semantics”: e.g., Fodor, Lepore and Capellen, Borg *inter alia*. Lexical semantics = Tarski like clauses for denotation. No interesting lexical semantics whatsoever.
- syntax semantics interface approaches (HPSG): specification of lexical entries in a typed unification based framework (TFSs). Well developed but limited.
- Decompositional approaches (Dowty 1979, Jackendoff 1990). Problems with decomposition (*kill*, *cause to die*) and with axiomatizing or giving a precise meaning to the primitives used.
- Generative Lexicon (Pustejovsky 1995): collects intriguing empirical data but little attention given to the syntax/semantics interface or to compositional semantics, attendant formalization problems.

Some Basic Concepts

- Meaning as use vs. meaning as truth conditions (the conditions or situations in which a given sentence is true)
-
- Giving truth conditions in a logically perspicuous language (Tarski, Montague, Davidson), with a clear notion of truth or satisfaction and semantic consequence.
- Beginning or in parallel with a syntactic analysis, semanticists construct a *logical form* for a sentence or a discourse, a representation of the truth conditional meaning of the sentence or discourse.
- Sentence or discourse meaning primary. Lexical meaning derived from contributions to truth conditions and observed entailments.

More about Lexical Meaning

Word choices can give rise to inferences.

1. John is a bachelor $\longrightarrow$ John is unmarried and male.
2. John is unmarried $\not\longrightarrow$ John is unmarried and male.
3. Bill angered Fred $\longrightarrow$ Fred was angry.
4. The enemy sank the ship with a torpedo $\longrightarrow$ the ship sank / the torpedo sank the shi).

A note about words

What is a word? In some sense the answer is obvious: words are the things dictionaries define.
But words often come with inflection for case, number, gender, or with morphological affixes like the English (*-ion* : *afflict* $\rightarrow$ *affliction*, affecting meaning and syntactic category, and hence its combinatorial possibilities.

tools for lexical semantics and composition

λ calculus gives us a way of composing lexical entries.

- If $F(x)$ is a formula with x free, then $\lambda x.F(x)$ is a term of the λ calculus.
- Formulas are of type T, names are of type E
- Every word in the lexicon expresses (at least) one λ term.
- Combining terms: $\lambda x \phi(\alpha) \longrightarrow \phi(\frac{\alpha}{x})$ (β reduction, application)
- normalization: when all λ terms are reduced to standard formulas of the underlying language.

The typed λ calculus

- $\lambda x.\neg x(x)[\lambda x.\neg x(x)] \longrightarrow \neg(\lambda x.\neg x(x)[\lambda x.\neg x(x)])$
- complete normalization is not possible.
- introduce (well founded) types to enable normalization.
- A simple theory of types $(\mathrm{E}, \mathrm{T}, \rightarrow)$.
- every term must have a determinate type.
- the term predications in the above cannot be well-typed in the simple theory of types.
- $\lambda x\phi(\alpha) \longrightarrow \phi(\frac{\alpha}{x})$ only if the type of x and α match.

Is the simple theory of types given above sufficient?

Sortal restrictions: Not all predications are equal

1. Eats its breakfast the dog.
2. ?The dog contains a lot of information.
3. The dog is eating its breakfast.
4. The tree was slow.
5. The tree grew slowly.

Some observations

- Syntax partially determines well-formedness to predications, but
- verbs, adjectives, determiners, and adverbs have arguments, other words or groups of words they combine with, and these predicates come with restrictions on the semantic types of their arguments.
- copredication is a good test for type distinctness as predicates can shift and "adapt" to the type of their arguments:

1. John swept the kitchen.
2. John swept the dust.
3. #John swept the kitchen and the dust.

Different sorts

- abstract or informational objects vs. physical objects.
- facts vs. propositions
- animate vs. non animate objects
- locations and places vs. physical objects
- masses vs. countable objects
- eventualities vs. objects
- kinds vs. individuals

These types are commonly used in contemporary semantic analyses.

What are sorts?

- sortal theory: entailments, and part of "at issue content" (composes with various operators)
- Suppose *hit* has as a selectional restriction that both its internal and subject arguments must be physical objects:
  1. John didn't hit Mary $\rightsquigarrow \neg \mathrm{P}(j) \vee \neg \mathrm{P}(m) \vee \neg H(j,m)$
- some sort of not "at issue" content.

Presupositions

Projection behavior and semantic well-formedness

1. The present King of France could have been bald.
2. Is the present King of France bald?
3. The present King of France is not bald.
4. Sylvain's son is almost three years old.
5. If Sylvain has a son, then Sylvain's son is almost three years old.

"Projection" of sortal restrictions

1. ?The number two could have been red.
2. ?Is the number two soft?
3. ?The number two didn't hit Bill.
4. If numbers were physical objects, then the number two could have been red.

Other tests for presuppositions:

- the non-redundancy of presupposed content

Examples

- Two is an abstract object. Two is prime.
- Sylvain has a son. Sylvain's son is almost three years old.

- the inability to make certain discourse continuations on pre- supposed content, also impossible to make discourse continuations on type requirements

Examples

- After Susan got mad at John, she made up with him. #She (had) slapped him.

conclusions: sortal restrictions are type information

- selectional restrictions are not at issue content
- but also not just standard presupposed content.
- A different level of information from what's conveyed by a sentence.
- semantic anomalousness is different from necessary falsity.

Examples

- Lions are robots
- Lions are sets.

Selectional restrictions as type presuppositions

There are some parallels and advantages.

- the variability of the justification of presuppositions
  - presuppositions can be "bound" to or satisfied by existing at issue content.
  - at issue content can be "adjusted" so that it satisfies presuppositions (accommodation)
- evidence (see Asher *Lexical Meaning in Context*, Cambridge, 2011) that type presuppositions also have various ways of being justified.
- the dependence of presupposition justification on the discourse context

Relations between types

sub typing

An obvious relation between types

- $\text{LION} \sqsubseteq \text{ANIMAL} \sqsubseteq \text{PHYSICAL OBJECT} \sqsubseteq \text{E}$ (the domain of entities)
- $\text{NAT} \sqsubseteq \text{INT} \sqsubseteq \mathbb{Q} \sqsubseteq \mathbb{R}$
- $\text{E} \not\sqsubseteq (\text{E} \rightarrow \text{T})$
- $\text{E} \not\sqsubseteq \text{LION}$

Sub typing continued

Why sub typing is important for linguists

- a: $\lambda P \lambda Q \exists x(Px \wedge Qx)$
- cat: $\lambda u\ \text{cat}(u)$
- a cat: $\lambda Q\ \exists x(\text{cat}(x) \wedge Qx)$
- sleeps: $\lambda v\ \text{sleeps}(v)$
- a cat sleeps: $\exists x\ (\text{cat}(x) \wedge \text{sleeps}(x))$

Sub typing continued

In a typed λ calculus

- type on u should determine type of $\lambda u\, \mathrm{cat}(u)$
- type of $\lambda u \mathrm{cat}(u)$ will be the functional type from type(u) to the type of a formula (T).
- type of $\lambda u \mathrm{cat}(u)$ should justify or satisfy the selectional restriction of a—the type of P.
- If x: E, P: E $\rightarrow$ T, we can get a well typed, standard λ term for *a cat*
- and via normalization the desired formula for *a cat sleeps*.

Sub typing continued

So what are the types of

- u, P?
- answer is not so obvious

Do we need type identity for reduction, or rather subtype?

- thinking of selectional restrictions as type presuppositions: subtype suffices

Sub typing continued

Suppose

- $u : \text{CAT}$, $P : \text{E} \rightarrow \text{T}$?
- $\lambda u\, \text{cat}(u) : \text{CAT} \rightarrow \text{T}$
- notion of matching required for reduction: $\text{CAT} \rightarrow \text{T} \sqsubseteq \text{E} \rightarrow \text{T}$
- can we show this?
- we need a precise conception of $\sqsubseteq$.

Church-Montague Types

- types are interpreted set theoretically:
- E is the set of all entities.
- $\mathrm{E} \rightarrow \mathrm{T}$ is the set of all functions $f\colon \mathrm{E} \rightarrow \mathrm{T}$
- $\sqsubseteq = \subseteq$

Sub typing continued

- On the Church Montague conception,
  $\text{CAT} \rightarrow \text{T} \not\sqsubseteq \text{E} \rightarrow \text{T}$
- is u: CAT right?
- How can we tell?

Reconsidering the type of *cat*

- the type of u λu cat(u) should be the type presupposition of *cat*.
- what are the selectional restrictions of such a predicate?
- one way to answer this question is to look at:

Examples

- noun adjective combinations: # uncountable cat, fat cat
- predicative constructions: #A cat is a ZF set, cats are robots

- $u \not\sqsubseteq \textsc{Nat}, u \not\sqsubseteq \textsc{Ord}, u \sqcap \overline{\textsc{animal}} \neq \bot$
- $u \sqsubseteq \textsc{p}$ (physical object)
- $u : \textsc{p}$

Reconsidering the type of *cat*, continued

So...

- $\lambda u\, \mathrm{cat}(u) : \mathrm{P} \rightarrow \mathrm{T}$
- But still on the Church Montague conception of types, $\mathrm{P} \rightarrow \mathrm{T} \not\sqsubseteq \mathrm{E} \rightarrow \mathrm{T}$ (in fact these are disjoint sets)
- here is the putative type of physical first order properties: $\mathrm{P} \rightarrow \mathrm{T}$)
- here is the type of first order properties: $(\mathrm{E} \rightarrow \mathrm{T})$
- Shouldn't the type of physical first order properties be a subtype of the first order properties?
- Shouldn't physical first order properties be first order properties??
- Something has gone wrong...

A solution (Asher 2011)

- move to a proof theoretic conception of sub typing.
- $\alpha \subseteq \beta$ iff there is a proof from α to β in an appropriate proof system.
- change the type of P to involve a restricted quantification over types.
- $P\colon \Sigma x \sqsubseteq \mathrm{E}.x \to \mathrm{T}$
- assume $\mathrm{P} \sqsubseteq \mathrm{E}$ is an axiom of the proof system, which also contains a form of existential generalization for Σ and conditional proof for $\to$, and that $\sqsubset$ is transitive and reflexive.
- we can now prove $(\mathrm{P} \to \mathrm{T}) \vdash (\Sigma x \sqsubseteq \mathrm{E}.x \to \mathrm{T})$
- and so: $(\mathrm{P} \to \mathrm{T}) \sqsubseteq (\Sigma x \sqsubseteq \mathrm{E}.x \to \mathrm{T})$
- physical first order properties are first order properties.

Inference and subtyping

- subtyping relative to a type hierarchy (primitive subtype axioms) $\mathscr{T}$ and a proof relation defined over these.
- Application or β reduction revisited: $\lambda x \phi(\alpha) \longrightarrow \phi(\frac{\alpha}{x})$, provided $\mathscr{T} \vdash_{\mathscr{T}} \text{type}(\alpha) \sqsubseteq \text{type}(x)$
- **Type Inferences**
  Suppose ψ implies $F(t)$ where t has type a. Then ψ dynamically implies $F(t) \wedge \phi_\alpha(t)$, where ϕ_α is the logical form counterpart of the type α.
- *bachelor* $\longrightarrow \lambda x$: ADULT $\sqcap$ MALE Bachelor(x)
  Kim is a bachelor $\vdash$ Kim is male and adult

From Types to Entailments

Suppose for a type hierarchy $\mathscr{T}$, $\vdash_{\mathscr{T}} \alpha \sqsubseteq \beta$ Then:

- it is a conceptual truth that $\textsc{every}(\alpha, \beta)$
- it is an analytic truth that $\forall x(\phi_\alpha(x) \rightarrow \phi_\beta(x))$

Lexical inference is a matter of inferences over subtypes or over type incompatibilities (e.g., $\textsc{tuesday} \sqcap \textsc{monday}$)

Philosophical backdrop: two sorts of meanings

- types drive composition and are part of speaker competence.
- Externalist arguments show that truth conditional content is not determined by what's in the head.
- an external semantics (intensions) and an internal semantics (types), not part of truth conditional at-issue content.
- types are concepts not identified with sets of their inhabitants but rather proof objects with rules of application
- Category theory useful, because it allows us to characterize concepts without details of internal structure.

Detailing the proof conception of sub typing

- a proof theoretic conception of sub typing needs a proof theoretic framework.
- category theory
  - a system of types closed under $\rightarrow$ is a Cartesian Closed Category, and gives us other types as well ($\times$ product, $\vee$ disjoint union))
  - a Topos is a categorial model for a system of types with $\rightarrow, \Sigma, \Pi$ and furnishes other complex types (pullbacks or fibre products).
- modern type theories
  - inductive types, dependent types.

The categorial model

- Types as abstract objects with a proof theoretic meaning.
- Category theory will exhibit the proof meanings in a suitably abstract way.
- Categories: characterizing objects by their relations to other objects, by the morphisms they allow. (Crole 1993, Asperti & Longo 1991, Barr & Wells 1999)
- A very different picture from a set-theoretic model, where structure is represented internally (via the set-theoretic structure).

Some basic definitions

Definition

A *graph* consists of a set of objects (points) and a set of arrows between points (edges). In addition, there are two maps from Arrows to Objects, one labeled *Domain* and the other *Co-Domain*, which pick out the collections of first arguments and second arguments of arrows respectively.

Definition

A *deductive system* $\mathscr{D}$ is a graph in which for each object A in $\mathscr{D}$, there is an arrow 1_A, which is the identity map on A, and for each pair of arrows $f : A \to B$ and $g : B \to C$, $g \circ f : A \to C$, the composition of g with f, is defined. Objects can be identified with formulas and arrows with proofs. The closure under composition gives us a rule of inference.

Categories

Definition

A *category* is a deductive system where $\forall f : A \to B$, $g : B \to C$ and $h : C \to D$:

- $f \circ 1_A = f = 1_B \circ f$
- $(h \circ g) \circ f = h \circ (g \circ f)$

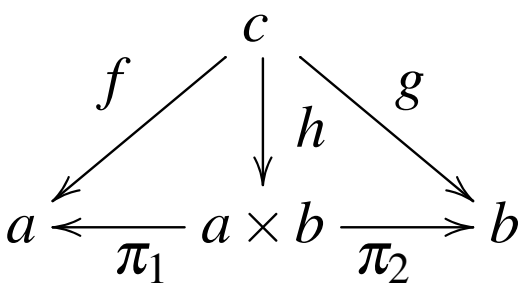
Categorial constructions

In category theory one can define certain constructions over objects like products in a category.

Definition

Let $\mathscr{A}$ be a category and a, b be objects in $\mathscr{A}$. Then $a \times b$ is the categorial product of a and b, which comes with two arrows $\pi_1 : a \times b \to a$ and $\pi_2 : a \times b \to b$ such that for any arrows $f : c \to a$ and $g : c \to b$ in $\mathscr{A}$ there is a unique $h : c \to a \times b$ such that $h \circ \pi_2 = g$ and $h \circ \pi_1 = f$.

A categorial product in a diagram all the arrows commute:



In the category of Set, product constructions correspond to Cartesian products.

Initial and terminal objects

Other constructions of interest in category theory are those of an *initial* and *terminal object*.

Definition

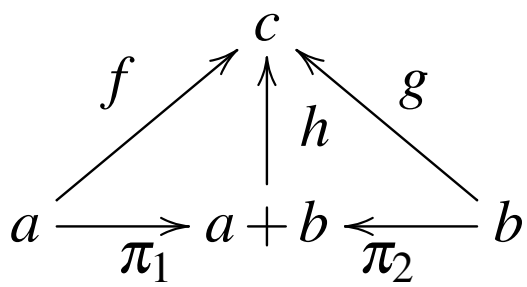
An object 0 is an initial object in a category $\mathscr{C}$ iff for all objects c in $\mathscr{C}$, there is a unique arrow $f: 0 \to c$ in $\mathscr{C}$. An object 1 is a terminal object in $\mathscr{C}$ iff for all objects c in $\mathscr{C}$, there is a unique arrow $f: c \to 1$ in $\mathscr{C}$.

Terminal objects are the duals of initials.

Duality

duality reverses the directions of the arrows in a construction

Categorial *coproducts* are the dual of products and correspond to disjoint union in set theory. Their commuting diagram:

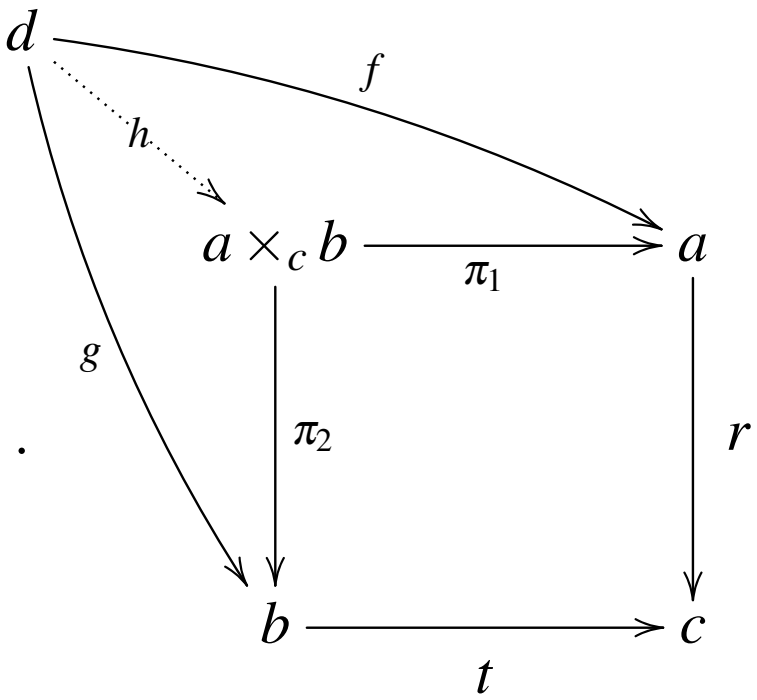


Another interesting construction, the pullback or fibre product

Definition

Let C be a category and let a, b, c be objects in $\mathscr{C}$, with morphisms $r : a \to c$, $t : b \to c$. The *pull back* with respect to r and t, denoted $a \times_c b$, is an object in $\mathscr{C}$ with two morphisms $\pi_1 : a \times_c b \to a$ and $\pi_2 : a \times_c b \to b$ satisfying $t \circ \pi_2 = r \circ \pi_1$, such that for any $d \in \mathscr{C}$ and morphisms $f : d \to a$ and $g : d \to b$ satisfying $t \circ g = r \circ f$, there exists a unique morphism $h : d \to a \times_c b$ such that $f = \pi_1 \circ h$ and $g = \pi_2 \circ h$.

A picture of a pull back



Embeddings

- If a category $\mathscr{C}$ admits of pull backs for all arrows r and t of $\mathscr{C}$, then it admits of products for all objects a, b of $\mathscr{C}$.
- In the category of set, a pull back forms sets of those pairs in a Cartesian product $a \times b$ that project down to the same element in c via the maps given from a to c and from b to c. It thus yields a set of equivalence classes of pairs.

Categories with more structure

A *cartesian category* is one that contains a terminal object and that is closed under the product construction. Duality ensures that a cartesian category also contains coproducts and initial objects.

Definition

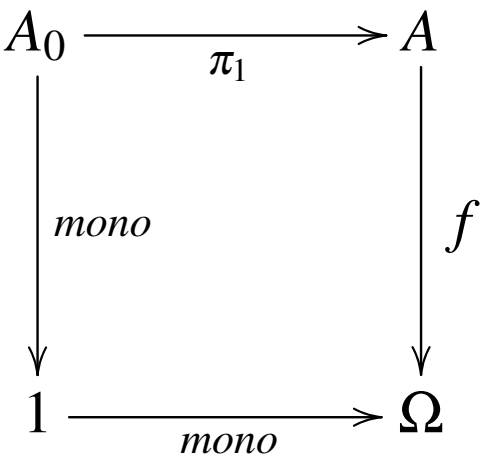
Let $\mathscr{C}$ be a cartesian category with objects a, b in $\mathscr{C}$. The *exponent* of a and b is object b^a in C together with a map $\text{EVAL}: b^a \times a \to b$ such that for all maps $f: c \times a \to b$ there exists a unique $h: c \to b^a$ such that $(h \times \text{ID}) \circ \text{EVAL} = f$. Note that $h \times \text{ID}: c \times a \to b^a \times a$.

Exponent objects of the form b^a have a domain
$\text{Do}(b^a) = a$ and a co-domain $\text{Co}(b^a) = b$.
A *cartesian closed category*, or *CCC*, is a cartesian category that is closed under the exponent operation.

Topoi

- A *topos* is a CCC with a subobject classifier . A subobject of an object A is a collection of monomorphisms $f_0 : A_0 \to A$ such that $f_0 = f_1$, where $f_1 : A_1 \to A$ iff $\exists$ isomorphisms $h : A_0 \to A_1,\ g : A_1 \to A_0$ such that $f_1 \circ h = f_0$ and $f_0 \circ g = f_1$.
- think of a subset A_0 of A as an embedding $m : A_0 \to A$.
  The subobject classifier Ω with 0 and 1 as subobjects and f unique

.



- A topos is closed under pull backs and under power objects—i.e. for any object A $[A \to \Omega]$ is the power object of A.

Categories and Logic

- A CCC is a model for the positive implicational fragment of intuitionistic logic (Lambek and Scott 1986)
- A CCC with a non empty object E is a model of the simple theory of types (we set 1 and 0 to be the Boolean values of the type T; exponent represents the operation of λ abstraction, EVAL the operation of application; the equality axioms of Andrews's Q_0 formulation of ST hold in a CCC.)
- A topos is a model for the higher order theory of types. Quantifiers over types and connectives are definable within a topos (Goldblatt 1979).

Montague Grammar in category theory

A good way to understand the power of category theory is to link it to a familiar formal semantics, MG (Lambek 1969, Pollard 2011). Pollard's site: http://www.ling.ohio-state.edu/ pollard/lacl6/prop-corrected.pdf
Montague's categorial grammar is an example of a biclosed monoidal category. A monoidal category with an object 1 and an associative operation $\otimes: \mathfrak{C} \times \mathfrak{C} \rightarrow \mathfrak{C}$, in which objects include not only basic objects but also morphisms. Thus if a and b are objects, $a \rightarrow b$ is also an object.
A closed monoidal category $\mathfrak{C}$ has a right adjoint a^{-1} for each object a in $\mathfrak{C}$ (takes a into 1 but can also take a into another object b).
$N \rightarrow NP$ is then an adjoint to N that yields an NP in the presence of an N; it is isomorphic to $N^{-1} \otimes NP$.
An expression of syntactic type A is a morphism

$$1 \rightarrow A.$$

On an analogy to A as a set we can identify a unique morphism from $1 \rightarrow A$ for each element of A.

Syntactic derivations

a	cat	sleeps
$(1 \to (N \to NP))$	$\otimes(1 \to N)$	$\otimes(1 \to (NP \to S)$
	$(1 \to NP)$	$\otimes(1 \to (NP \to S))$
	$1 \to S$	

From syntax to semantics

- Every BMC can be embedded in a CCC and every CCC can be embedded in a topos.
- More complex categories simply add more structure, just as groups can be embedded within rings, or arithmetic within analysis.
- Thus each syntactic category is mapped into a semantic type via this morphism $f : \mathfrak{C}_{bmc} \rightarrow \mathfrak{C}_{\tau}$.
- An object of syntactic type A, i.e. a morphism $s\colon 1 \rightarrow A$ is mapped via f to a semantic value of type B, $\sigma : 1 \rightarrow B$.
- The translation from syntactic derivations to semantic values takes place via this embedding.

Topoi and semantic values

- A Cartesian Closed Category furnishes the basics, providing a model of $\Rightarrow$, $\vee$ and functional and disjunctive types.
- Consider the ordinary way of thinking of a semantic value of a 1 place predicate as a function from individuals into $\{1,0\}$ (the characteristic function of the set denoted by the predicate).
- Lambek's proposal: identify sentence meanings with $1 \rightarrow \Omega$. This yields only a purely extensional theory. Only two propositions!

Rethinking semantic values categorically

- types are concepts not identified with sets of their inhabitants but rather something like proof objects with rules of application
- We distinguish a Pre Boolean Algebra object PROP distinct from Ω. PROP$/\equiv$ is a BA, where $\equiv$ is the equivalence relation induced on PROP by the usual Boolean identities. (Pollard 2011)
- PROP is inductively constructed from a basic set of subtypes of E, not to be counfounded with their extensions. Boolean operations and standard quantifiers can be defined within the topos.
- Ω plays the counterpart of Montague's type T.

Fine grained intensionality and proof objects

- Category theory is ideal for characterizing types/concepts, because it allows us to characterize them without details of internal structure.
- Can use $\equiv$ to provide BA-like equivalences over properties as well as propositions without collapsing concepts that are equivalent.

From categories back to proof interpretations

- types as proof theoretic objects:
- a simple type A gives us instructions for finding morphisms $1 \rightarrow A$:
- $\alpha \rightarrow \beta$ a proof given something x of type α yields a proof that x is of type β.
- more complex type constructors in terms of their morphisms from category theory.

Subtyping defined

Definition

Δ is the smallest set of formulas such that if $\alpha \sqsubseteq_{\mathscr{T}} \beta, \alpha \rightarrow \beta \in \Delta$

Definition

$\vdash_\Delta$ derivations: contain no introduction of theorems on lines and

$$\frac{\Delta,\ \alpha \vdash \beta \text{ and } \Delta \nvdash \beta}{\alpha \vdash_\Delta \beta}$$

Definition

Subtyping: $\alpha \sqsubseteq \beta$ iff $\alpha \vdash_\Delta \beta$

Commentary

- The notion of subtyping needed a bit of work.
- The notion of subtype is “overloaded": it is to be linked to deduction, but it is also supposed to play its traditional role in the type hierarchy.
- $\sqsubseteq$ is not a type constructor, but a relation between types.
- nevertheless, it has a meaning close to that of $\rightarrow$.

An alternative from Modern Type Theory

Coercive sub typing

a is a subtype of b if a's "can be seen in a given context" as b's.

- $a \leq_c b$ iff there is an arrow from a into b such that $c[a]$ is an internal object of b
- generalization of subsumptive subtyping.
- Coherence and conservatively require coercions to be unique: $a \leq_c b$ and $a \leq_{c'} b \rightarrow c = c'$.
- Luo and Soloviev (2012) show that coercive sub typing is conservative over the basic type system MTT, which is consistent relative to ZF.

Some more on MTT

- MTT is a mono stratal theory: all semantics done via proofs and types.
- MTT does not have quantificational types, but it does have product and disjoint union types.
- coercive subtyping gives contravariance: $a \leq a' \vdash a \to b \leq a' \to b$.
- thus if common nouns were predicates, the type of *animal* would be a subtype of the type of *dog*.
- $\text{DOG} \leq \text{LION} \vdash \text{ANIMAL} \to \text{T} \leq \text{DOG} \to \text{T}$
- MTT's solution: all common nouns are types, but verbs take predicates
- adjectives are dependent types that combine with common noun primitive types.
- to express determiners/noun combinations, we need determiners to select for universes (not really types). (See Lungo & Luo, LACL 2014)

Some other alternatives

Bekki & Tanaka (2014)

- : dependent types used for existential and universal quantifiers
- a man: $\Sigma x\colon \mathrm{E}\mathrm{man}(x)$,
- *a man* has a type that corresponds to a pair consisting of $t : \mathrm{E}$ and a proof that $\mathrm{man}(t)$.
- *a man enters*: $\Sigma u\colon (\Sigma x\colon \mathrm{E}\mathrm{man}(x))\mathrm{enters}(\pi_1(u))$
- see also Type Theory with Records (TTR) (Cooper, Ginzburg)

Rules for type composition

- TCL uses simply typed lambda calculus (relaxed reduction, substitution).
- but like continuation style semantics uses a type to pass typing requirements from predicates to arguments
- a list of types on arguments.
- requires rules for type presupposition satisfaction.

Binding Type Presuppositions

Definition

$$\frac{\gamma \sqsubseteq \alpha,\ \mathrm{ARG}_i^P : \alpha,\ \mathrm{ARG}_i^P : \gamma)}{\mathrm{ARG}_i^P : \gamma}$$

Sometimes presuppositions can't be bound.

Definition

Simple Type Accommodation

$$\frac{\alpha \sqcap \beta \neq \bot,\ \mathrm{ARG}_i^P : \alpha * \mathrm{ARG}_i^P : \beta)}{\mathrm{ARG}_i^P : \alpha \sqcap \beta}$$

Determiners have type requirements that accommodate

Examples

I'll have a Chardonnay.

The type presupposition of the determiner is COUNT, but *Chardonnay* is neither MASS nor COUNT (or has a default for mass.
We get a modification of the noun by the determiner.

A derivation

- *Heavy*: $\lambda P\colon 1\ \lambda x\colon \mathrm{E}\ \lambda \pi''\ (P(\pi'')(x) \wedge \mathrm{heavy}(x, \pi'' * \mathrm{ARG}^{\mathrm{heavy}}\colon \mathrm{P}))$
- *tree*: $\lambda \mathscr{P} \lambda x \lambda \pi\ \mathscr{P}(\pi * \mathrm{arg}_1^{\mathrm{tree}}\colon \mathrm{P})(x)(\lambda v \lambda \pi' \mathrm{tree}(v, \pi'))$
- Applying the lexical entry for *tree* to that for *heavy*, we obtain:
  $\lambda z \lambda \pi \lambda P \lambda x \lambda \pi''(P(\pi'')(x) \wedge\ \mathrm{heavy}(x, \pi'' * \mathrm{ARG}_1^{\mathrm{heavy}}\colon \mathrm{P}))$
  $(\pi * \mathrm{ARG}_1^{\mathrm{tree}}\colon \mathrm{P})(z)(\lambda u \lambda \pi'\ \mathrm{tree}(u, \pi'))$
- After several uses of Substitution and Application:
  $\lambda z \lambda \pi\ (\mathrm{tree}(z, \pi * \mathrm{ARG}_1^{\mathrm{tree}}\colon \mathrm{P}) \wedge\ \mathrm{heavy}(z, \pi * \mathrm{ARG}_1^{\mathrm{tree}}\colon \mathrm{P} *$
  $\mathrm{ARG}_1^{\mathrm{heavy}}\colon \mathrm{P}))$
- Binding and Substitution reduce the presuppositional contexts:
  $\lambda z \lambda \pi\ (\mathrm{tree}(z, \pi * \mathrm{ARG}_1^{\mathrm{tree}}\colon \mathrm{P})\ \wedge\ \mathrm{heavy}(z, \pi * \mathrm{ARG}_1^{\mathrm{tree}}\colon \mathrm{P}))$

Now for a derivation that doesn't work

Above, we had two compatible typings on the same variable. When we attempt the same derivation for the modification *heavy number*, we get an irresolvable type clash in the presuppositions.

Examples

$\pi * \text{ARG}_1^{\text{number}} : \text{ABSTRACT} * \text{ARG}_1^{\text{heavy}} : \text{P}.$

We cannot justify the type presuppositions of the noun and the adjective with any of our rules. So the derivation crashes and no well-formed lambda term corresponds to this noun phrase and it has no proof-theoretic interpretation.

Categorial models for TCL

Fact

Application, Abstraction, and hence Substitution are all sound in a CCC.

Interpret Application via Eval, Abstraction via the unique function h associated with a exponent object. Substitution is the identity rule.
The rule from From Subtying to Logic yields a reasonable logical relation from $\sqsubseteq$. In particular,

Fact

Subtyping is sound in a topos.

Fact

Simple Accommodation is sound in a topos.

Fact

Rules for existential quantification are sound in a topos.

5 problems for lexical semantics

- types and selectional restrictions $\surd$ (types as presuppositions)
- subtyping $\surd$ (restricted inference relation)
- dual aspect terms or terms with multiple types.
- coercion
- shifts of lexical meaning in discourse contexts

Dual aspect nouns and complex types

- featured in Cruse (1986), and taken up in Pustejovsky (1995).
- little details of how aspects are exploited in the semantics.

Examples

Ducks lay eggs and are common to most of Europe. [*individual* and *kind*]

Snow is common this time of year and all over my back yard. [*kind* and *mass*]

Mary picked up and mastered three books on mathematics. [*physical object* and *informational content*]

More examples of "dot" types

- Je ne suis qu'un roseau mais je suis un roseau pensant. (Pascal) [*physical object* and *thinking agent*]
- That is a lump of bronze but also Bernini's most famous famous statue [*portion of matter* and *artifact*
- Le prix Goncourt is 10000 euros and a great honor not accorded every year. [*amount of money* and *prize*)
- The lecture (interview, speech) lasted an hour and was very interesting. (*event* and *information*)
- The bank is just around the corner and specializes in sub prime loans. (*physical object/ location* and *institution*).
- The canvas is immense, and, as an example of a bygone school, interesting to all art lovers. (*New York Times* March 24, 1894) [*physical object* and *informational object*]

Copredication, counting and quantificational shift with dual aspect nouns

quantificational puzzles

shifts in quantificational domains

Examples

John has read every math book in the library. John has mastered every math book in the library. John has stolen every math book in the library

Examples

John answered every question. John repeated every question.

Verbs can select aspects of nouns that force shifts in the domain.

Analyzing dual aspect nouns and copredication

- simple ambiguities don't support copredication

Examples

The bank is slippery from the recent rains and specializes in IPOs.

- copredications are ellipses; copy the DP with the dual aspect noun into the second predication and disambiguate. (predicts false readings for quantified sentences like *Mary carried home and then mastered 3 books on semantics*)
- dual aspect nouns have a conjunctive type: $\text{BOOK} \sqsubseteq \text{I} \sqcap \text{P}$.
  (but no object is both an individual instance of a kind and a kind...)
- dual aspect nouns have a product type:
  $\text{BOOK} \sqsubseteq \text{I} \times \text{P}$
  Better, but...

Counting arguments

a. there are exactly one copy combining *War and Peace* and *Anna Karenina* two copies of *Ulysses*, and six copies of the *Bible* on a shelf .

b. Pat has read *War and Peace* and *Ulysses*, and no other book

c. Sandy has read the *Bible* (and looked at all 6 versions), and no other book.

Now, consider the questions:

- (Q1) How many books are there on the shelf?
- (Q2) How many books has Pat read?
- (Q3) How many books has Sandy read?
- (Q4) Who has read more books, Pat or Sandy?

Most people would answer:

- 9 to (Q1)
- 2 to (Q2)
- 1 to (Q3)
- Pat to (Q4)

Pairing approach gives us too many objects

- types are pair types defined by a left and right projection.

The pairing approach gives us 10 pair objects consisting of a physical object represented by a number of the physical copy and an information content represented by the title of the book or an abbreviation thereof:

- (1, W&P), (1, AK), (2, U), (3, U), (4, Bible), (5, Bible), (6, Bible),(7, Bible), (7, Bible), (8, Bible), (9, Bible).

- Dual aspect nouns require a special type structure.
- Asher (2011) models this as a pull back.

Upshot: dual aspect nouns require a special type

Need a local variable that's of type P and one of type I that are linked to *book*

- BOOK $\sqsubseteq$ I $\bullet$ P
- model this type as sui generis but licensing a natural transformation to a fibre product, which says, roughly, that given a maps from one aspect to another, there exists an object that has these two aspects.
- copredication involves a type justification in which an aspect type is selected, and this may affect the domain.
- the selection of an aspect type will determine counting and individuation principles and will shift domains of the quantifiers accounting for the quantificational puzzle.

Some counting pictures

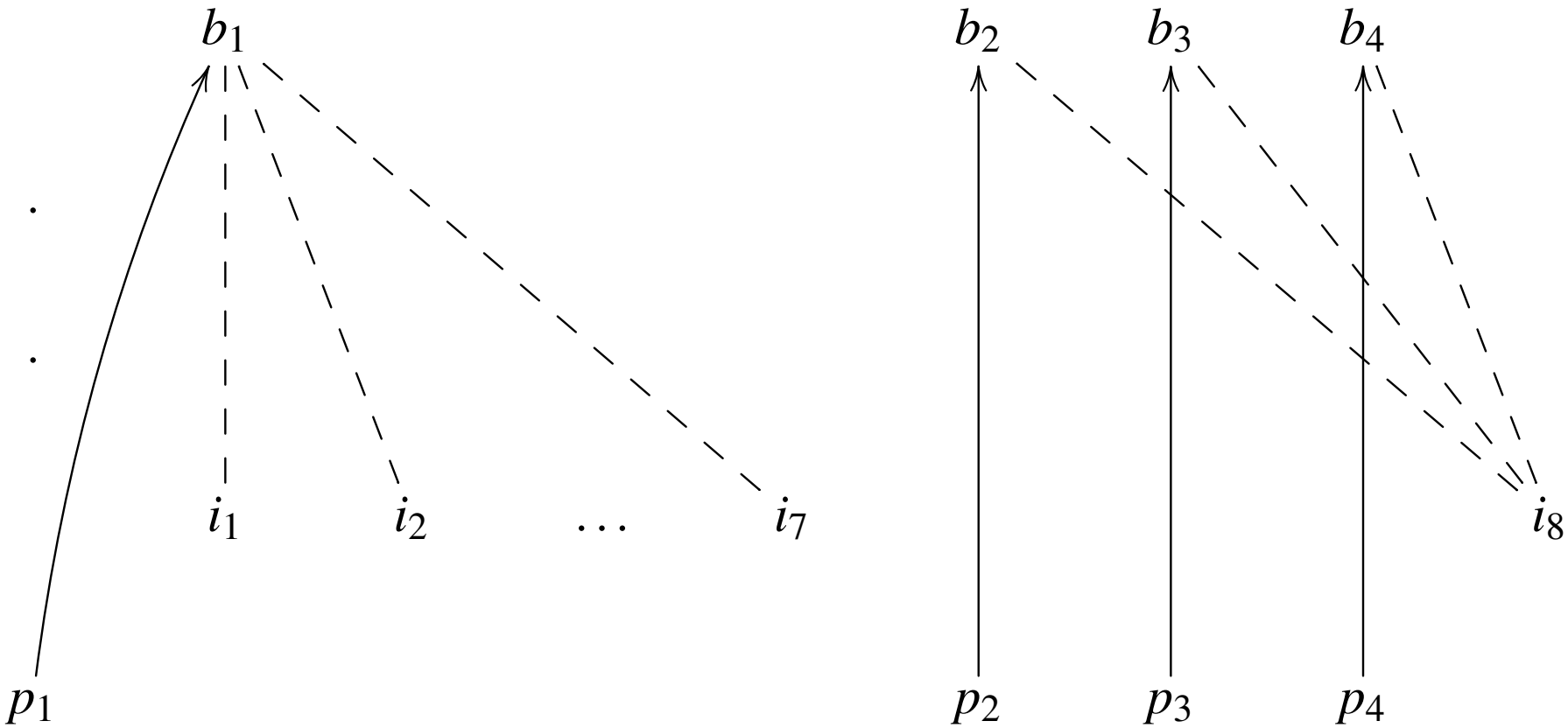


Figure: Books individuated physically

Books individuated informationally

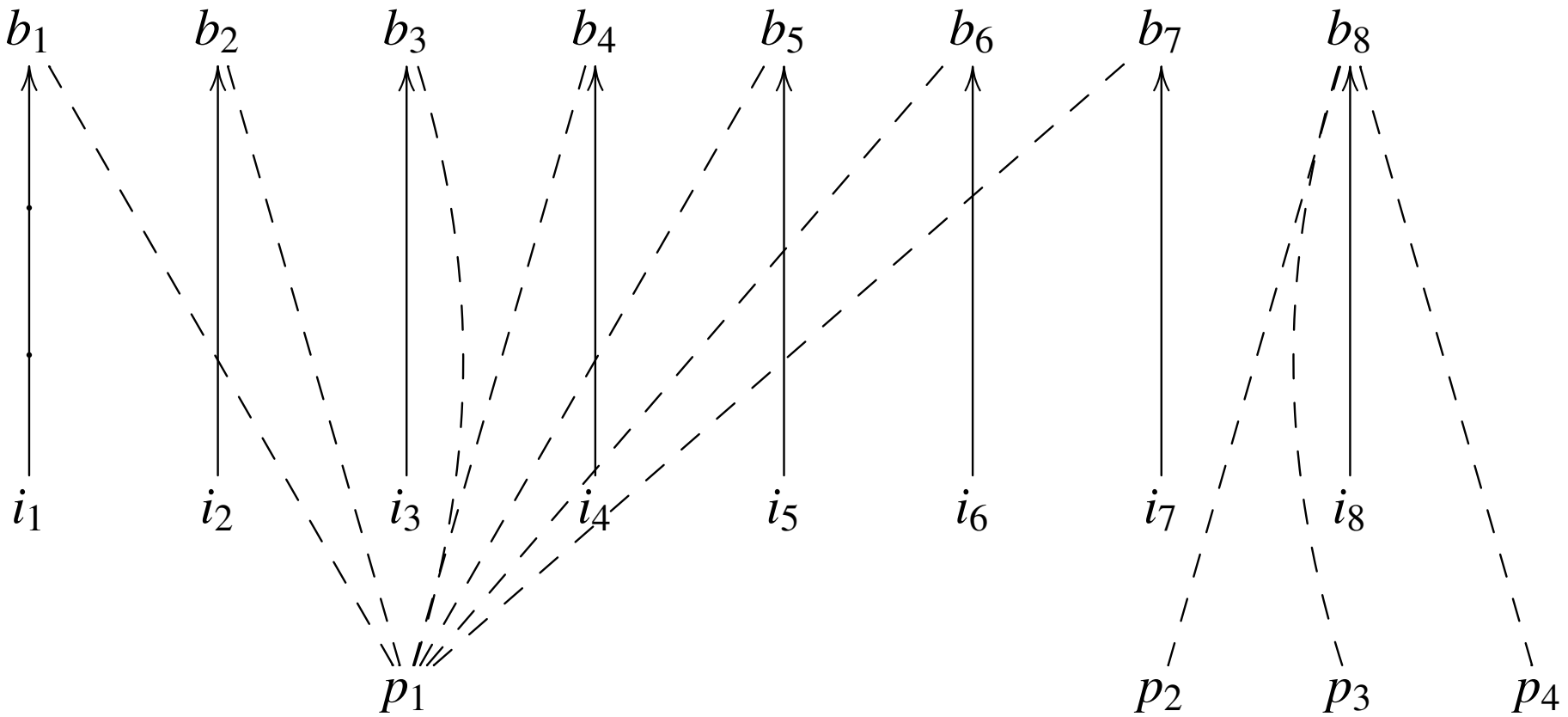


Figure: Books individuated informationally

Category theoretic implementation

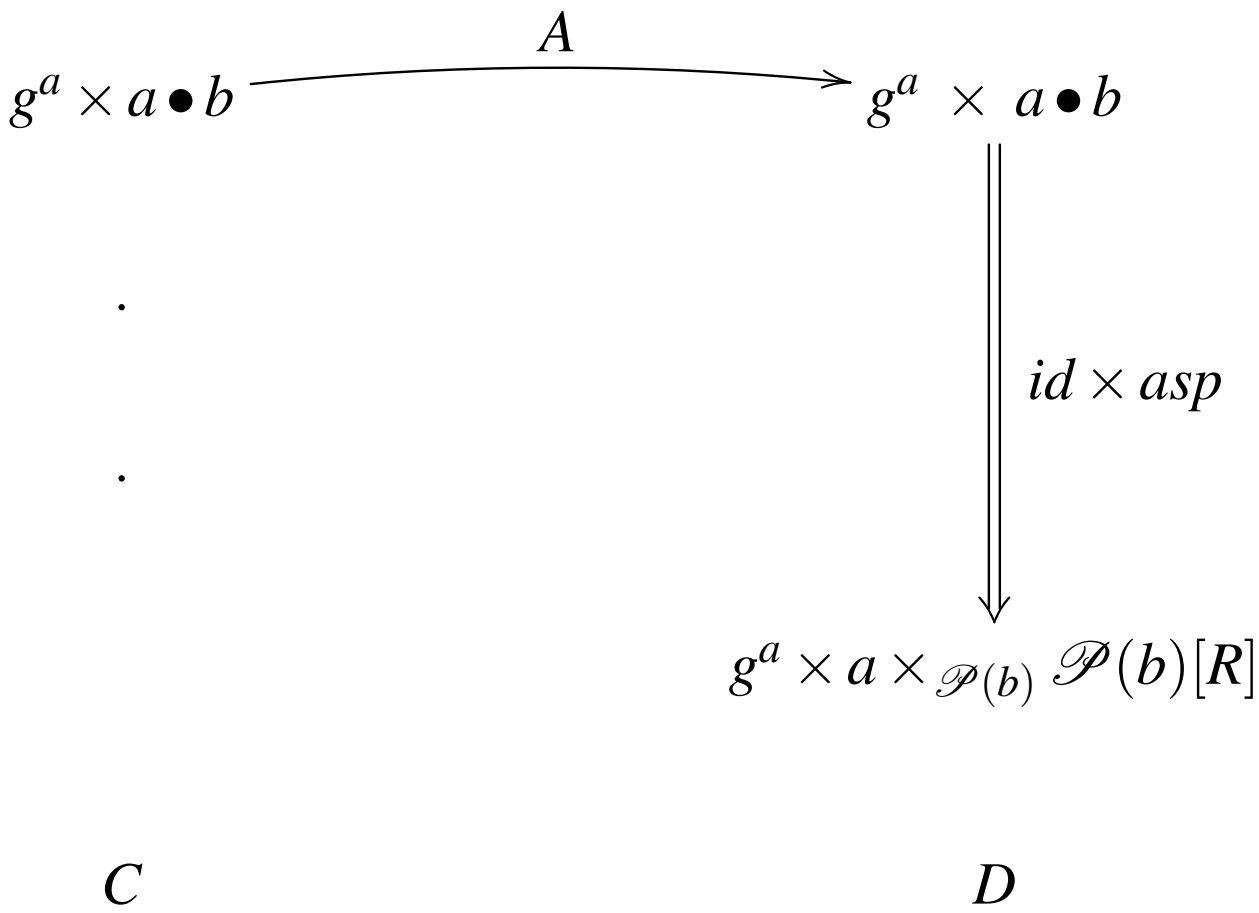
I distinguish a particular class of transformations, I'll call them *Aspect* or *A*, from one topos *C* that has • types to another *D* containing a new arrow from the bullet type to the pullback of the right sort. (analogous to type raising but more like a horizontal shift)

Definition

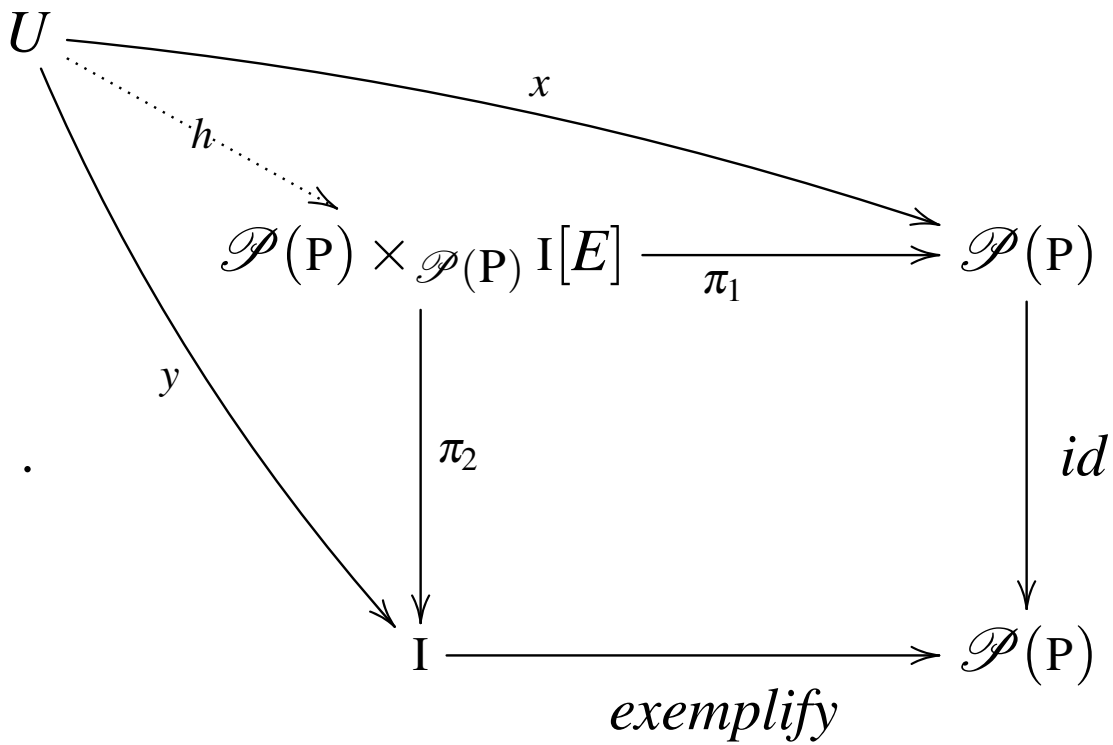
$\alpha \times \beta[R]$ is the restriction of the product/pullback $\alpha \times \beta$ such that for the projections π_1 and π_2 of $\alpha \times \beta$ $\pi_1(x) = y$ and $\pi_2(x) = z$ iff $R(y,z)$

Transformations

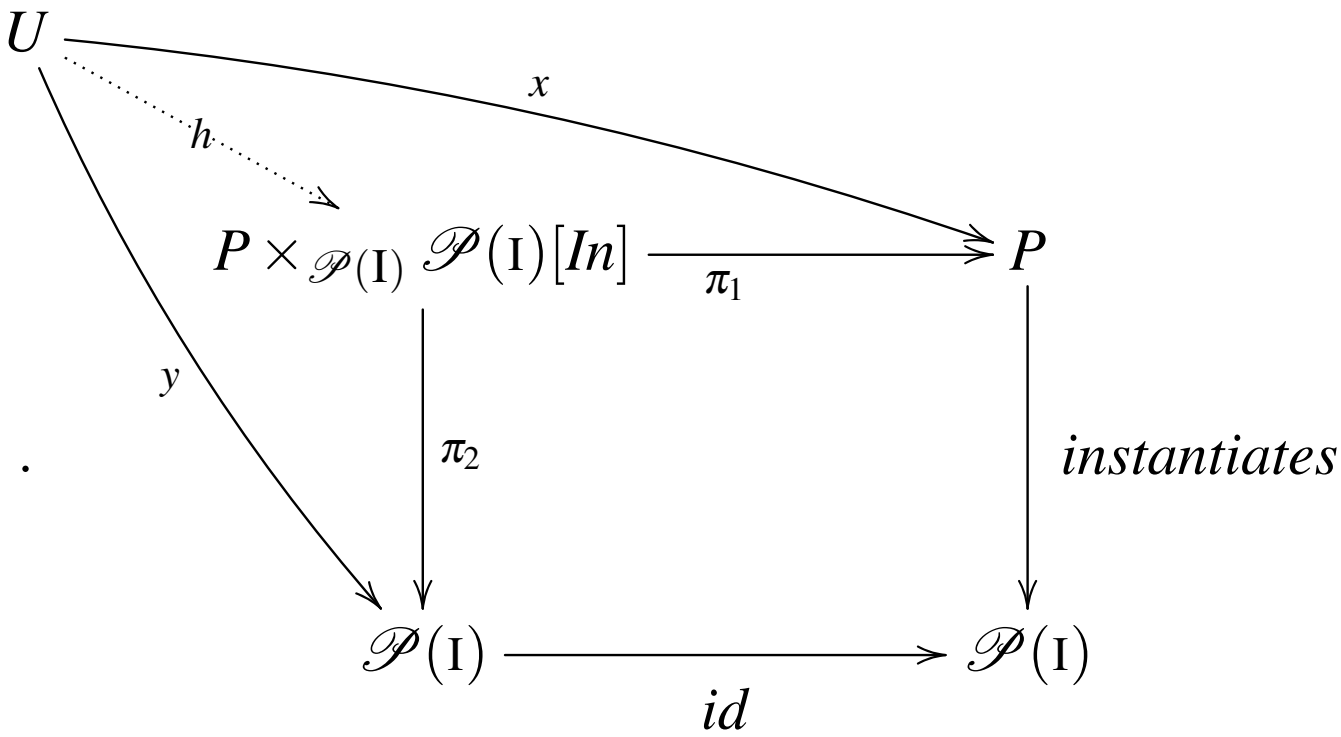
We can now graph the natural transformation A and its associated arrow, *aspect*.



The "I" pullback



The "P" pullback



Upshot of this transformation in TCL

- a new variable is created for the aspect in the transformation or aspect selection.
- *heavy book* $\lambda x\colon \mathrm{P} \bullet \mathrm{I} \exists y\colon \mathrm{P}(\ \text{o-elab}(y,x) \wedge \text{heavy}(y) \wedge \text{book}(x))$
- aspect selection is a kind of presupposition justification and it applies globally if possible.
- *John stole 3 books*: $\exists_3 z\colon \mathrm{P}\ \exists x\colon\ :\ \mathrm{P} \bullet \mathrm{I}\ (\text{pick-up}(j,z) \wedge\ \text{o-elab}(z,x))$
- in copredication, however, global justification of two incompatible aspects is not possible and so we have as expected with presupposition a local justification.
- John picked up and mastered three books.
- $\exists_3 w\colon \mathrm{P} \bullet \mathrm{I}\ (\text{book}(w) \wedge \exists z\colon \mathrm{P}\ (\text{pick-up}(j,z)$
  $\wedge\ \text{o-elab}(z,w)) \wedge \exists z'\colon \mathrm{I}\ (\text{master}(j,z') \wedge\ \text{o-elab}(z',w))))$

Dual aspect nouns with coercive subtyping

- the pairing hypothesis doesn't lead to the same conclusion because of the limiting scope of the coercion.
- You coerce either to p-books or to i-books but not both within the same scope (coercions must be unique).
- Also terms outside the scope of the coercion preserve their original type.
- but how do quantificational shifts work; is the determiner in the scope of the coercion? If only the noun, then looks like we can't recover the original type though we get quantificational shifts.

What do dot types do to our ontology?

- in MTT, the question of subsumption is complex, because of the presence of coercions.
- if you assume certain coercions you get one type hierarchy, (e.g., $a \times b \leq_{\pi_1} a$) and if you assume others you get another $a \times b \leq_{\pi_2} b$
- in TCL, you have the simple type hierarchy and then dot types are *sui generis* but provably subtypes of E.

Thinking more about sub typing

- subsumptive sub typing is natural metaphysically (identity map for coercive subtype).
- coercive sub typing much more general and more permissive, but how do we interpret it metaphysically— a Hans Vaihinger neo Kantian sub typing? treat α as if it were β within the context C.