Extended abstracts of the ESSLLI 2015 workshop TYTLES: Types Theory and Lexical Semantics

Robin Cooper, Christian Retoré

Extended abstracts of the ESSLLI 2015 workshop

TYTLES: Types Theory and Lexical Semantics

Workshops organizers / programme committee co-chairs:
Robin Cooper & Christian Retoré

Barcelona, 3-7 August 2015

Those are the extended abstracts of the TYTLES (Barcelona, August 3-7 2015) workshop together with the slides that were projected during the introduction and the conclusion.

Some authors may publish alternative versions later on elsewhere.

Program committee:

- Robin Cooper (University of Gothenburg, Co-Chair),
- Christian Retoré (Université de Montpellier, & LIRMM Co-Chair)
- Alexandra Arapinis (CNR, Trento)
- Nicholas Asher (CNRS, Toulouse)
- Christian Bassac (Université Lyon II)
- Stergios Chatzikyriakidis (CNRS et LRIMM, Montpellier)
- Shalom Lappin (King's College, London)
- Zhaohui Luo (Royal Holloway, University of London)
- Chiara Melloni (CNR, Verona)
- Bruno Mery (Université de Bordeaux)
- Richard Moot (CNRS, Bordeaux)
- Glyn Morrill (Universitat Polytècnica de Catalunya, Barcelona)
- Larry Moss (Indiana University, Bloomington)
- Reinhard Muskens (Universiteit Tilburg)
- Livy Real (IBM Research, Saõ Paolo)

Presentation

The pioneering work of Ranta (1994) on using Type Theory for NL semantics has initiated a strong interest in the use of Type Theories for representing formal semantics. And even though Type Theory was initially mainly concerned with compositional and formal semantics, a number of linguists, logicians and computer scientists noticed the relevance of type theory for lexical semantics as well. Around 2000 the paper "the metaphysics of words in context" by Asher & Pustejovsky (2001) initiated Type Theoretic approaches to lexical coercions and meaning transfers by investigating extension and refinement of the type system used by Montague. Accounts for this type of phenomena need to capture ordinary phenomena of selection restriction (e.g. a "chair" may not "bark", in an ordinary context), while at the same time they have to ensure some flexibility for adapting meanings to contexts in case of meaning transfers, co-predication etc. The study of this kind of phenomena is of course not new. Their study goes back at least till the 80's (Bierwisch, Nunberg, Cruse among others). What is relatively new is the study of these phenomena from the perspective of Type Theory and this approach is by now quite successful as valuable type theoretical contributions on incorporating lexical considerations into compositional semantics show (Asher, Bassac, Chatzikyriakidis, Cooper, Luo, Melloni, Mery, Moot, Prévot, Pustejovsky, Ranta, Real, Retoré). The topics of the workshop include:

- Linguistically motivated variants of type theories (subtyping)
- Lexical semantics in type theory (compositionality and the lexicon)
- Interaction between lexical semantics and type theoretical semantics
- Classical semantic questions in richly typed frameworks (plurals, quantification, generics)
- Modelling specific questions in type theory (nouns, deverbals, events, adjectives, adverbs, ontological aspects,)
- Computational aspects and implementation of type theoretical semantics (natural language inference, proof assistants…)

Some references:

- Asher, N. (2011), *Lexical Meaning in Context: A Web of Words*, Cambridge University Press.
- Asher, N. & Luo, Z. (2012), Formalisation of coercions in lexical semantics, *in* 'Sinn und Bedeutung 17'.
- Asher, N. & Pustejovsky, J. (2005), 'Word Meaning and Commonsense Metaphysics', in course materials for Type Selection and the Semantics of Local Context, ESSLLI 2005.
- Bassac, C.; Mery, B. & Retoré, C. (2010), 'Towards a type-theoretical account of lexical semantics', *Journal of Logic, Language and Information* **19**(2), 229--245.
- Chatzikyriakidis, S. & Luo, Z. (2013), Adjectives in a Modern Type-Theoretical Setting In Morrill, G. & Nederhof, M.-J. *Formal Grammar, LNCS 8036 pp. 159-174.*
- Chatzikyriakidis, S. & Luo Z. (2014). Natural Language Reasoning Using Proof Technology: Rich Typing and Beyond. *Proceedings of EACL2014*
- Cooper, R. (2011), Copredication, Quantfication and Frames, *in* S. Pogodalla & J.-Ph. Prost, ed., 'Logical Aspects of Computational Linguistics: 6th International Conference, LACL 2011', Springer, pp. 64--79.
- Cooper, R.; Dobnik S.; Lappin, S.; Larsson, S. (2014) A Probabilistic Rich Type Theory for Semantic Interpretation. ACL Workshop on Type Theory and Natural Language Semantics, Göteborg, 2014.
- Cooper, R. (2010), Type Theory and Semantics in Flux, *In* Kempson, R.; Asher, N. & Fernando, *T. Handbook of Philosophy of linguistics. pp. 271-323.*
- Jezek, E. & Melloni, Ch. (2011) Nominals, polysemy and copredication, *Journal of cognitive sciences* **12** pp**.** 1-31.
- Moot, R.; Prévot, L. & Retoré, C. (2011) A discursive analysis of itineraries in an historical and regional corpus of travels: syntax, semantics, and pragmatics in a unified type theoretical framework In *Contraints in Discourse*
- Ranta, A. (1994), *Type-Theoretical Grammar*, Clarendon Press, Oxford.
- Real L. & Retoré, C. (2013), Deverbal semantics and the Montagovian generative lexicon. *Journal of Logic Language and Information. 2014* DOI: 10.1007/s10849-014-9187-y
- Retoré, C. The Montagovian Generative Lexicon ΛTyn: a Type Theoretical Framework for Natural Language Semantics in Ralph Matthes and Aleksy Schubert (eds) 19th International Conference on Types for Proofs and Programs (TYPES 2013) LIPICS vol 26 pp. 202--229.

Table of contents / program

(by order of the talks, 3 per session 14:00-15:30 from Monday August 3 to Friday August 7 2015)

*Introductio*n **1**
Christian Retoré (and Robin Cooper)

A Puzzle about Long-distance Indefinites and Dependent Type Semantics **13**
Justyna Grudzinska and Marek Zawadowski.

Type Theories and Lexical Networks: **21**
Using Serious Games as the Basis for Multi-Sorted Typed Systems
Stergios Chatzikyriakidis, Mathieu Lafourcade, Lionel Ramadier and Manel Zarrouk.

Perceptual Meaning in TTR Judgement-based Semantics and Conceptual Spaces.....**27**
Staffan Larsson.

Interfacing Language, Spatial Perception and Cognition in Type Theory with Records **34**
Simon Dobnik.

Probabilistic Mereological TTR and the Mass/Count Distinction **41**
Peter Sutton and Hana Filip.

Are Widows Always Wicked? Learning concepts through enthymematic reasoning **49**
Ellen Breitholtz.

The Relative Complexity of Constraints in Co-Predicative Utterances **56**
Bruno Mery.

Calculating Projections via Type Checking **62**
Daisuke Bekki and Miho Satoh.

Quantification in Frame Semantics with Hybrid Logic **69**
Laura Kallmeyer, Timm Lichte, Rainer Osswald, Sylvain Pogodalla and Christian Wurm.

An Overview on Portuguese Nominalisation **77**
Livy Real and Alexandre Rademaker.

Formalising type-logical grammars in Agda **83**
Pepijn Kokke.

A Developed Analysis of Type Coercion Using Asher's TCL and Conventionality **91**
Seohyun Im and Chungmin Lee.

Factivity and Presupposition in Dependent Type Semantics **100**
Ribeka Tanaka, Koji Mineshima and Daisuke Bekki.

Final summary and discussion: **108**
emerging themes in type theory and lexical semantics (from the above articles)
Robin Cooper (and Christian Retoré).

TYTLES
TYpes Theory and LExical Semantics: Introduction to the Workshop

Robin Cooper University of Gothenburg
Christian Retoré Université de Montpellier LIRMM

ESSLLI 2015 Barcelona

1. Lexical semantics

Usually lexical semantics refers to:

- Word meaning in context (various forms of polysemy)
- Relation between meanings
- Lexical networks

Remark: relation between meanings $\rightarrow$ higher order logic is needed

Usual techniques:

- description with features (human / non human)
- sometimes arguments structures specifying the nature of the arguments
- in NLP: word vectors

2. Polysemy

(1) Simple
   a. The river passed the bank.
   b. The bank is nearby the river.
   c. The bank phoned me.

(2) Institutions and such
   a. The journal is printed on pink paper.
   b. The journal hired a new commentator.
   c. The journal is nearby the port.

(3) Events
   a. The signature took three months.
   b. The signature is unreadable.

(4) Dot objects
   a. The book is red.
   b. The book is quite interesting.

3. Borderline examples

(5) a. I am parked behind a blue BMW.
   b. The ham sandwich asked for a coffee.

Rather supports contextualism in the radical minimalism / contextualism debate.

4. Copredication

(6) Dinner was delicious but took ages. (event / food)

(7) * The salmon we had for lunch was lightning fast. (animal / food)

(8) I forgot on the table my preferred book on logic. (physical / info)

(9) I carried the books from the shelf to the attic since i already read them. (phys. / info)

(10) Liverpool is a poor town and an important harbour. (people / geographic)

(11) * Liverpool defeated Chelsea and is an important harbour. (football / geographic)

(12) (context: ok) Barcelona won four champions leagues and organised the olympiads.

(13) (contrast: ok) Libourne, a small south-west town, defeated Lille.

5. Integrating lexical semantics into a compositional and computational framework

Standard lexical semantics, distributional semantics (in NLP big data, machine learning):

what a text speaks about

Formal semantics:

what a (few) sentence(s) assert(s)

6. Complementary approaches

(14) (was geach a student of Wittgenstein) In 1941, he married Elisabeth Anscombe, by whom he got in contact with Wittgenstein. Although he never attended his lectures, he was strongly influenced by him.

(15) The children will have a pizza.

Both word meaning and sentence/discourse structure are needed to understand.

In Man Machine Interaction no large data but proper understanding is mandatory.

7. Pustejovsky's generative lexicon 1/4

Pioneering work:

- compositional (generative) view of word meaning
- formal framework : word meaning as complex feature structures the way they combine is less specified
- can be implemented for computing semantics
- qualia structure can be learnt (at least partly)

8. Pustejovsky's generative lexicon 2/4

The four levels of an entry:

- lexical typing structure: giving an explicit type for a word positioned within a type system for the language;
- argument structure: specifying the number and nature of the arguments to a predicate;
- event structure: defining the event type of the expression and any subeventual structure it may have; with subevents;
- qualia structure: a structural differentiation of the predicative force for a lexical item.

9. Pustejovsky's generative lexicon 3/4

Qualia Structure:

- formal: the basic category of which distinguishes the meaning of a word within a larger domain;
- constitutive: the relation between an object and its constituent parts;
- telic: the purpose or function of the object, if there is one;
- agentive: the factors involved in the object's origins or "coming into being.

10. Pustejovsky's generative lexicon 4/4

Types in Pustejovsky:

- base types organised as an ontology
- there also are functional types
- in the argument structure the types are specified

11. Pustejovsky's GL: conclusion

Compositional semantics that integrates lexical aspects.

Entries are well defined.

The base types ans their ontology remain obscure.

The composition modes are described on examples there is no general procedure that could be implemented, and the relation to syntax is no explicited.

Is this framework able to compute the semantics of a whole complex sentence, or of a small discourse?

Can one learn other levels than qualia structure?

12. Restrictions of selection

A common way to start addressing lexical issues in compositional semantics:

(16) The chair barked.

(17) Dictionary: "barks" is said from an animal, usually a dog.

Simple and good idea: infelicitous semantic composition is type mismatch.

A function/predicate is expecting an argument of type A and receives an argument of type B.

Such constraint needs to be relaxed because of the context.

(18) I was so late for registration that the secretary barked at me.

13. Lexical issues are idiosyncratic

Observe that meaning transfers are idiosyncratic:

(19) a. Ma voiture est crevée.
b. My can has a pinhole?
c. My car is flat?

(20) a. My brother in law is trivial.
b. He always makes trivial jokes.
c. A trivial vector space.
d. Apart from the trivial solution $x^2 + x$ has no real solution.

14. Four frameworks: 1/3 TCL

TCL: type composition logic (Asher)

Simply typed lambda calculus with many types.

Meaning : formulae of higher order logic

Enriched with composition rules that import constructions from category theory (e.g. pullbacks for dot object).

Question: do the terms obtain from lexical entries by those enriched rules always reduce to a meaningful term/formula?

15. Four Frameworks: 2/4 MTTc+ DTS

Asher & Luo, Bekki, Chatzikyriakidis, Mineshima, Meaning logic and composition logic: type theory (Subtyping)

Dependent types: family of types $B(a)$ with x in a type A (dependent types may depend on terms).

Type $\Pi a : A.B(a)$ types for functions that maps a proof of A into a proof of $B(a)$ such functions may also be viewed as a term/proof of $\forall a : AB(a)$.

Type $\Sigma a : AB(a)$ types of couples (a, b) with a proof/term of type A and a proof/term of type $B(a)$ such couples may also be viewed as a term/proof of $\exists a : AB(a)$.

Coercice subtyping: $$\frac{f : A \to B \quad a : A' \quad A' <_c A}{f(a) : B}$$

Here: Seohyun Im and Chungmin Lee — Daisuke Bekki and Miho Satoh — Ribeka Tanaka, Koji Mineshima and Daisuke Bekki.

16. Four frameworks: 3/4 TTR

Type theory with records Cooper

Meaning logic and composition logic: type theory

Record types are sequences of types terms $t_i : T_i$ where T_{i+1} may depend (dependent type) on the typed terms t_k with $1 \leq k \leq i$.

$$\begin{bmatrix} \text{x} & : & \textit{Real} \\ \text{loc} & : & \textit{Loc} \\ \text{e} & : & \text{temp(loc, x)} \end{bmatrix}$$

$$\begin{bmatrix} \text{scale} & : & (\textit{AmbTempFrame} \rightarrow \textit{Real}) \\ \text{e} & : & \textit{AmbTempFrame}^2 \\ \text{c}_{\text{rise}} & : & \text{scale(e[0])} < \text{scale(e[1])} \end{bmatrix}$$

Here papers by : Staffan Larsson — Simon Dobnik — Peter Sutton and Hana Filip — Ellen Breitholz — Pepijn Kokke

17. Four frameworks: 4/4 MGL

Montagovian Generative Lexicon (Retoré, Mery, Bassac, Moot, Real ...) As Montague or TCL: typed lambda calculus for composing minings expressed in a different language (higher order predicate logic).

Montague with several base types (like Musken's Ty_n)

Quantification over types to factor uniform combinations.

Coercions are word specific (and not ontological)

For copredication some coercions block others.

Here: Bruno Mery

18. Common question 1) "base types"

What should be the base types?

- a dozen of ontological types (Asher, TCL)
- classifiers (like in Japanese, Mandarin, Sign Languages ...)
- common nouns (Luo)

A hint can be provided by dictionaries: how do they express the restriction of selection? which classes/sorts/types do they use?

19. Common question 2) subtyping

Related to base types: if many of them subtyping is needed for natural coercions (=? ontological inclusions)

Subtyping with quantification over types: complicated! The $\forall\alpha$ should preserve subtyping for positive occurrences of the type variable α and reverse subtyping for negative occurrences of α.

As said above for MTT a good solution by Luo & Soloviev: coercive subtyping Technical requirement that makes it work: at most one coercion between any two types. Adaptation to System F by Lang & Retoré

A question does linguistic generalisation correspond to ontological generalisation ?

20. Common question 3) learning

Machine learning techniques cannot learn sophisticated structures like types and terms. An exception without sequel: Luke S. Zettlemoyer and Michael Collins. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. (2005)

An alternative exploiting existing lexical resources (Here: Stergios Chatzikyriakidis, Mathieu Lafourcade, Lionel Ramadier and Manel Zarrouk. — Livy Real and Alexandre Rademaker)

- Turning existing data (e.g. lexical networks like WordNet, or JeuxDeMots crowdsourced by an Internet game) into the terms and types we need.
- Groups of words that are similar in some aspect and the aspect which makes them similar as well can be observed.
- Privileged relation can be extracted as well (e.g. is an agent, an object a location for).

Another alternative (Cooper, Here: Ellen Breitholtz):

- Mimicking the interactive process by which we do learn meanings.
- Allow every one to have his one lexicon,
- Lexicons may evolve.
- Quite interesting *per se* about human cognition.

21. Common question 4) revisiting compositional semantics

Having a complicated type system allow to revisit standard issue in formal semantics:

- quantification (in situ quantification with Hilbert's epsilon works much better in the typed case) Here: Justyna Grudzinska and Marek Zawadowski — Laura Kallmeyer, Timm Lichte, Rainer Osswald, Sylvain Pogodalla and Christian Wurm
- plurals (Mery, Moot, Retoré)
- count/mass nouns (Mery, Moot, Retoré Here: Peter Sutton and Hana Filip)
- predicates/type judgements (Luo, Retoré) $a : A$ vs. $\tilde{A}(a)$.

A Puzzle about Long-Distance Indefinites and Dependently Typed Semantics with Generalized Quantifiers

Justyna Grudzińska[1] and Marek Zawadowski[2]

[1] University of Warsaw, Institute of Philosophy
[2] University of Warsaw, Institute of Mathematics

Abstract. Indefinites (e.g. *a woman*, *some problem*) have given rise to a number of puzzles concerning their scopal and dynamic behavior. One such puzzle about long-distance indefinites seems to be unsettled in the literature ([2], [14], [13]). In this paper we show how the puzzle of long-distance indefinites can be handled in Dependently Typed Semantics with Generalized Quantifiers (DTSGQ). The proposal builds on our formal system combining generalized quantifiers ([9], [5], [1]) with dependent types ([8], [11], [7], [6]) in [3].

Keywords: quantifier scope, long-distance indefinite, dependent type

1 Background

The puzzle about long-distance indefinites arises in connection with sentences such as

(1) Every linguist has studied every solution that some problem might have.

Sentence (1) allows the so-called long-distance intermediate scope reading saying that for every linguist there is a possibly different problem such that he/she has studied every solution that this problem might have. This reading is considered exceptional, for the indefinite *some problem* takes scope out of its syntactic island (unlike standard quantifiers). Kratzer in [4] credits the problematic reading to the presence of a hidden pronoun/functional element, i.e. the apparent long-distance intermediate readings are in fact bound/functional readings and they only become available when there is a contextually salient pairing each of the linguists with some particular problem/function present. The intended readings can be expressed by the following paraphrases

(1a) Every linguist has studied every solution that a certain problem that intrigued him/her might have.

(1b) Every linguist has studied every solution that some problem that intrigued him/her most might have.

To capture the readings, Kratzer uses the mechanism of 'Skolemized choice functions (CF)'. Chierchia in [2] observes that there is a second kind of long-distance readings that cannot be reduced to bound/functional readings

(2) Not every linguist has studied every solution that some problem might have.

Sentence (2) is intuitively true in a situation where: for some linguist there is no problem such that he/she has studied every solution that this problem might have. These are the truth-conditions for the negated long-distance intermediate reading: for every linguist there is some problem such that he/she has studied every solution that this problem might have (and not for the negated Kratzer's bound/functional reading). Chierchia's proposal is to capture the long-distance intermediate reading using the mechanism of the intermediate existential closure of the CF variable. So the puzzle is that we need two mechanisms to account for the behavior of long-distance indefinites: Skolemized CF (as pointed out by Schlenker in [12], Skolemized CF are needed to account for some clear-cut cases of functional readings) and the intermediate existential closure of the CF variables. Moreover, the two mechanism are problematic on both theoretical and empirical grounds (see e.g. [10]).

2 Our Proposal

Our Dependently Typed Semantics with Generalized Quantifiers (DTSGQ) combines two semantic approaches to account for natural language quantification: Generalized Quantifier Theory familiar from Montague-style semantics ([9], [5], [1]) and type-theoretic approach ([8], [11], [7], [6]). Like in the classical Montague-style analysis, DTSGQ makes essential use of generalized quantifiers (GQs). But in the spirit of the type-theoretic framework we adopt a many-typed analysis (in place of a standard single-sorted analysis). Like in the standard type-theoretic approaches, we have type dependency in our system. But our semantics is model-theoretic (with truth and reference being basic concepts), and not proof-theoretic (where proof is a central semantic concept).

Combining GQs with dependent types allows us to handle in a uniform manner a number of semantic puzzles concerning natural language quantifiers. In our previous work we have defined a new interpretational algorithm to account for a wide range of anaphoric (dynamic) effects associated with natural language quantification ([3]). In this paper we will show how the puzzle of long-distance indefinites can be handled in DTSGQ. We propose to credit the problematic readings to the presence of (possibly hidden) dependencies or functions, i.e. the apparent long-distance intermediate readings involve in fact either dependent types or functions.

2.1 Dependently Typed Semantics with Generalized Quantifiers (DTSGQ)

In this and the following sections, we only discuss the elements of the system relevant for the linguistic purposes of this paper. For the full system, see [3].

Polymorhic interpretation of quantifiers. Standard Montague-style semantics is single-sorted in the sense that there is a single type e of all entities. On the Montague-style analysis, quantifiers are interpreted over the universe of all entities E. Our semantics is many-sorted in the sense that there are many types and we have a polymorphic interpretation of quantifiers. On the Montague-style analysis, quantifier phrases, e.g. *some woman*, are interpreted as sets of subsets of E

$$\|\exists x : woman\ x\| = \{X \subseteq E : \|W\| \cap X \neq \emptyset\}.$$

On our analysis, a generalized quantifier associates to every set Z a subset of the power set of Z: $\|Q\|(Z) \subseteq \mathcal{P}(Z)$; quantifier phrases, e.g. *some woman*, are interpreted as follows

$$\|\exists_{w:Woman}\| = \{X \subseteq \|W\| :\ X \neq \emptyset\}.$$

As a consequence of our many-typed analysis, predicates are also defined polymorphically, i.e. predicates are interpreted over many types (and not over the universe of all entities).

Combining quantifier phrases. To handle multi-quantifier sentences, the interpretation of quantifier phrases is further extended into the interpretation of (generalized) quantifier prefixes. (Generalized) quantifier prefixes can be built from quantifier phrases using the sequential composition ?|? constructor. The corresponding semantical operation is known as iteration (see [3]). To illustrate with an example: *Every linguist has studied some problem* can be understood to mean that each of the linguists has studied a potentially different problem. To capture this reading:

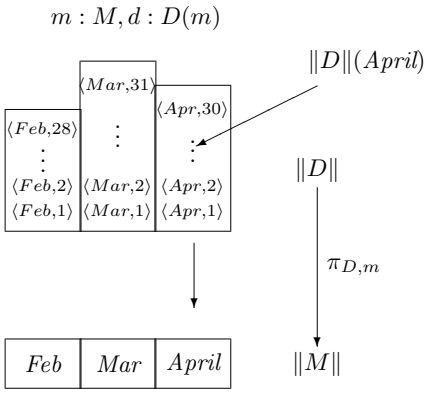
- a sequential composition constructor ?|? is used to produce a multi-quantifier prefix: $\forall_{l:L}|\exists_{p:P}$;
- the corresponding semantical operation of iteration is defined as follows

  $$\|\forall_{l:L}|\exists_{p:P}\| =$$

  $$\{R \subseteq \|L\| \times \|P\| : \{a \in \|L\| : \{b \in \|P\| : \langle a, b\rangle \in R\} \in \|\exists_{p:P}\|\} \in \|\forall_{l:L}\|\}.$$

The multi-quantifier prefix $\forall_{l:L}|\exists_{p:P}$ denotes a set of relations such that the set of linguists such that each linguist is in this relation to **at least one problem** is the set of all linguists. Obviously, the iteration rule gives the same result as the standard nesting of quantifiers in first-order logic.

Dependent types. Crucially, in a system with many types we can also have dependent types. One example of such a dependence of types is that if m is a variable of the type of months M, there is a type $D(m)$ of the days in that month



If we interpret type M as a set $\|M\|$ of months, then we can interpret type D as a set of the days of the months in $\|M\|$, i.e. as a set of pairs

$$\|D\| = \{\langle a, k\rangle : k \text{ is (the number of) a day in month } a\},$$

equipped with the projection $\pi_{D,m} : \|D\| \to \|M\|$. The particular sets $\|D\|(a)$ of the days of the month a can be recovered as the fibers of this projection (the preimages of $\{a\}$ under $\pi_{D,m}$)

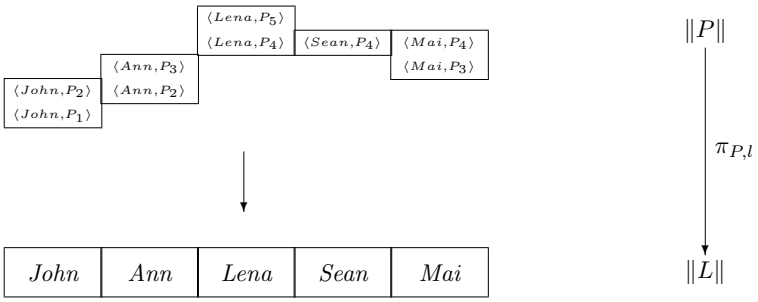
$$\|D\|(a) = \{d \in \|D\| : \pi(d) = a\}.$$

Generalized quantifiers on dependent types. Generalized quantifiers are extended to dependent types in our system

$\|\forall_{l:L}|\exists_{p:P(l)}\| =$

$\{R \subseteq \|P\| : \{a \in \|L\| : \{b \in \|P\|(a) : \langle a, b\rangle \in R\} \in \|\exists_{p:P(l)}\|(\|P\|(a))\} \in \|\forall_{l:L}\|\}.$

The multi-quantifier prefix $\forall_{l:L}|\exists_{p:P(l)}$ denotes a set of relations such that the set of linguists such that each linguist is in this relation to **at least one problem in the corresponding fiber of problems** is the set of all linguists. By extending the interpretation of generalized quantifiers to dependent types, our semantics introduces quantification over fibers, e.g. quantification over the fiber of the problems of John - $\|P\|(John)$



2.2 DTSGQ Analysis of Sentence (1)

Alphabet. The alphabet of the system consists of:
type variables: $X, Y, Z, \ldots$;
type constants: $Linguist, Problem, Solution, \ldots$;
type constructor: $\mathbb{T}$;
individual variables: $x, y, z, \ldots$;
predicates: $P^n, P^n_1, \ldots$;
quantifier symbols: $\exists, \forall, \ldots$;
prefix constructors: $?|?, \ldots$.

English-to-formal language translation. Consider now a sentence in (1):

(1) Every linguist has studied every solution that some problem might have.

Our English-to-formal language translation process consists of two steps (i) *representation* and (ii) *disambiguation*. The syntax of the representation language - for the English fragment considered in this paper - is as follows

$S \rightarrow Prd^n(QP_1, \ldots, QP_n)$;
$MCN \rightarrow Prd^n(QP_1, \ldots, CN, \ldots, QP_n)$;
$MCN \rightarrow CN$;
$QP \rightarrow Det\ MCN$;
$Det \rightarrow every, some, \ldots$;
$CN \rightarrow linguist, problem, \ldots$;
$Prd^n \rightarrow study, have, \ldots$

Sentence (1) is accordingly represented as

$Study^2$ (*every linguist*, *every solution that some problem might have*).

Multi-quantifier sentences of English, contrary to sentences of our formal language, are often ambiguous. Hence one sentence representation can be associated with more than one sentence in our formal language. The second step thus involves disambiguation. We take quantifier phrases out of a given representation and organize them into possible prefixes of quantifiers. In the case of our example, the sentence translates as

$$\forall_{l:Linguist}|\forall_{t_s:\mathbb{T}_{Solution\ to\ some\ problem}} Study^2(l, t_s).$$

Interpretation. In the Montague-style semantics, common nouns are interpreted as predicates (expressions of type $e \to t$). In our type-theoretic setting, common nouns (CN), e.g. *linguist*, are interpreted as types; modified common nouns (MCN), e.g. *solution that some problem might have*, are treated as $*$-*sentences* ($= Have^2$ (*some problem*, *solution*) determining a system of (possibly dependent) types, and the types so determined are interpreted using an interpretational algorithm defined in [3]. In the case of our example, the type *Linguist* is interpreted as a set of linguists (indicated in the context) $\|Linguist\|$, and the type $\mathbb{T}_{Solution\ to\ some\ problem}$ is interpreted as

$$\|\mathbb{T}_{Solution\ to\ some\ problem}\| =$$

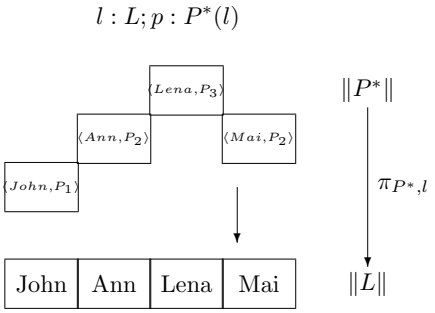
$$\{c \in \|Solution\| : \{b \in \|Problem\| \ : \langle b, c\rangle \in \|Have\|\} \in \|\exists_{p:Problem}\|\}.$$

Thus, as can be seen from this analysis, DTSGQ can only yield a narrow scope reading for the indefinite *some problem* in (1): every a in $\|Linguist\|$ has studied every c in $\|\mathbb{T}_{Solution\ to\ some\ problem}\|$.

2.3 DTSGQ Solution to the Puzzle about Long-Distance Indefinites

Indefinites. Unlike standard quantifier expresions, indefinites have been claimed to be ambiguous between a quantificational and a referential reading. Our proposal ties this ambiguity to the variability in type assignment. A quantificational indefinite *a/some problem* combines a determiner *a/some* and the variable of the type *Problem*, interpreted as the set of all problems (given in the context) - $\|Problem\|$. A referential indefinite *a/some (certain) problem* combines the determiner with the variable of the referential type *Problem**, interpreted as a certain singleton set consisting of a problem that the speaker has in mind - $\|Problem^*\|$. Correspondingly to referential types, we can also have dependent referential types in our semantics.

Dependent and functional readings. Our proposal distinguishes dependent referential and functional readings. If a sentence like (1) involves a hidden dependent referential indefinite, e.g. *a (certain) problem (that intrigues him/her)*, then it quantifies over the dependent referential type:



yielding the dependent referential reading saying that every linguist a in $\|L\|$ has studied every solution that a certain (one) problem b in $\|P^*\|(a)$ might have. That is, DTSGQ gives a reading: every a in $\|L\|$ has studied every c in $\|\mathbb{T}_{\textit{Solution to a (certain) problem (that intrigues him/her)}}\| =$

$\{c \in \|S\| : \{b \in \|P^*\|(a) \ : \langle b, c\rangle \in \|Have\|\} \in \|\exists_{p:P^*(l)}\|(\|P^*\|(a))\}.$

If a sentence like (1) involves a hidden function inducing element (e.g. the *most intriguing problem* function), then we get a functional reading saying that every linguist a has studied every solution to $f(a)$.

If sentence (1) involves a quantificational indefinite, DTSGQ does not give a long-distance intermediate (Chierchia's) reading (as explained above, DTSGQ can only yield a narrow scope reading for the indefinite). Chierchia's reading, however, can be explained away. As observed by Chierchia in [2], special context is needed to get a long-distance intermediate reading for (1) (in the absence of factors inducing dependent referential or functional readings), e.g. 'You know, linguists are really systematic: Lee studied every solution to the problem of weak crossover, Kim every solution to the problem of donkey sentences, etc.' We propose that people posit certain dependencies (given some such context), e.g.: that the type of problems depends on the type of linguists and the type of solutions depend on the type of problems $l : L; p : P(l); s : S(p)$ - by quantifying over the so posited dependent types, we get the apparent dependent ($\simeq$ Chierchia's) reading

$\forall_{l:L}|\exists_{p:P(l)}|\forall_{s:S(p)} Study^2(l, s)$

(for every linguist a in $\|L\|$ there is a problem b in $\|P\|(a)$ such that a has studied every solution c in $\|S\|(b)$).

The negative sentence (2) would then claim that for some a in $\|L\|$ there is no problem b in $\|P\|(a)$ such that a has studied every solution c in $\|S\|(b)$.

Acknowledgments

The work of Justyna Grudzińska was funded by the National Science Center on the basis of decision DEC-2012/07/B/HS1/00301. The authors would like to thank the anonymous reviewers for their valuable comments.

References

1. Barwise, J., Cooper, R.: Generalized Quantifiers and Natural Language. Linguistics & Philosophy 4, 159-219 (1981)
2. Chierchia, G.: A Puzzle about Indefinites. In Cecchetto C., Chierchia G., and Guasti M.T. (eds.), Semantic Interfaces: Reference, Anaphora, and Aspect. CSLI, Stanford, 51-89 (2001)
3. Grudzińska, J., Zawadowski, M.: System with Generalized Quantifiers on Dependent Types for Anaphora. In Cooper, R., Dobnik, S., Lappin, S., Larsson, S. (eds.), Proceedings of the EACL 2014 Workshop on Type Theory and Natural Language Semantics, 10-18 (2014).
4. Kratzer, A.: Scope or Pseudoscope? Are there Wide-Scope Indefinites? In Rothstein S. (ed.), Events and Grammar, Kluwer Academic Publishers, Dordrecht, 163-196 (1998).
5. Lindström, P.: First-order predicate logic with generalized quantifiers. Theoria 32, 186-95.(1966)
6. Luo, Z.: Formal Semantics in Modern Type Theories with Coercive Subtyping. Linguistics & Philosophy 35, 491-513 (2012)
7. Makkai, M.: First Order Logic with Dependent Sorts, with Applications to Category Theory. Preprtint McGill University (1995)
8. Martin-Löf, P.: An intuitionstic theory of types. Technical Report, University of Stockholm (1972)
9. Mostowski, A.: On a generalization of quantifiers. Fundamenta Mathematicae 44, 12-36 (1957)
10. Onea, E.: Why indefinites can escape scope islands. Linguistics & Philosophy (to appear)
11. Ranta, A.: Type-Theoretical Grammar. Oxford University Press, Oxford (1994)
12. Schlenker, P.: A note on Skolem functions and the scope of indefinites. Poster, NELS (1998)
13. Szabolcsi, A.: Quantification. Cambridge University Press, Cambridge (2010)
14. Schwarz, B.: Two kinds of long distance indefinites. In Rooy R. van, Stokhof M. (eds.), Proceedings of the Thirteenth Amsterdam Colloquium, Amsterdam University, 192-197 (2001)

Type Theories and Lexical Networks: Using Serious Games as the Basis for Multi-Sorted Typed Systems.

Stergios Chatzikyriakidis, Mathieu Lafourcade, Lionel Ramadier and Manel Zarrouk*

LIRMM
University of Montpellier 2
stergios.chatzikyriakidis@lirmm.fr
mathieu.lafourcade@lirmm.fr
lionel.ramadier@lirmm.fr
manel.zarrouk@lirmm.fr

Abstract. In this paper, we show how a rich lexico-semantic network which has been built using serious games, *JeuxDeMots*, can help us in grounding our semantic ontologies as well as different sorts of information in doing formal semantics using modern type theories (type theories within the tradition of Martin Löf). We discuss the domain of base types, adjectival and verbal types, hyperonymy/hyponymy relations as well as more advanced issues like homophony and polysemy.

1 Introduction

Modern Type Theories, i.e. Type Theories within the tradition of Martin Löf [13,14], have become a major alternative to Montague Semantics (MS) in the last twenty years. A number of influential approaches using MTTs have been proposed by the years [15,11,16,7] and have shown that the rich typing system offered by these MTTs (type many-sortedness, dependent types, type universes) has considerable advantages over simple typed theories predominantly used in mainstream formal semantics. One further important aspect for considering the use of MTTs over traditional Montagovian frameworks concerns the proof-theoretic nature of the former but not of the latter.[1] This fact makes MTTs a suited formal semantics language to perform reasoning tasks, as these are exemplified for example in work on inference using proof-assistant technology [5,4]. However, this expresiveness of typing comes with a cost. For example, how does one decide on the base types to be represented? On the one hand, we do have a way to get a more fine-grained type system unlike the monolithic domain of

* This work is partially supported by the ANR ContInt Polymnie project in France.

[1] At least in the way it is employed in the Montagovian setting, simple type theory can be viewed as model theoretic. There is however, interesting work on the proof theory of simple type theory. The higher order theorem prover LEO-II [1] is an example of such work. We are grateful to an anonymous reviewer for pointing this out to us.

entities found in MS, but on the other hand, constructing such a type ontology is not at all straightforward and easy task. Different approaches and assumptions have been put forward. For example [11,12] proposed to treat CNs as types, in effect every CN is a different type. Approaches like [16] on the other hand, take a more moderate view by building type ontologies according to classifier systems (in effect type ontology is built using intuitions from classifier systems).

On the other hand, work with lexical-semantic networks have provided us with structured lexicons that indicate lexical and semantic relations like for example WordNet [8]. A very promising line of research in lexico-semantic network construction concerns networks which are built collaboratively by using GWAPs, games with a purpose i.e. serious games. This is the case of JeuxDeMots Lexical Network (JDM, [9]), which is constructed through many GWAPs along with a contributive tool (Diko) which allows players/users to contribute directly and to browse the knowledge base.

In this paper, we propose to ground our semantic ontologies as well as any other information needed in order to perform reasoning tasks using MTT semantics in the JDM lexical network. In order to do this, we present some first thoughts on how such an endeavour can be made by looking at the way a translation procedure from JDM to MTTs can be performed. Issues to be discussed include the domain of base types, instances of these types, adjectival and verbal types, hyponymy/hypernomy relations as well as more advanced issues like homophony and polysemy.

2 From JDM to MTTs

2.1 Base Types and Instances of Base Types

MTTs, as already said are many-sorted systems in that they involve a multitude of types rather than just one monolithic type e domain of entities. In the accounts proposed by [11,12], every CN is associated with a base type. Note the first major difference with MS: CNs are not predicates but rather Types.[2] For base types, an easy way to unify the two approaches is to assume that CNs are basically base types. In JDM, POS tagging will provide this sort of information. We further have to exclude instances of terms (for example $John$ as an instance of Man) in order to distinguish between instances of terms and terms themselves (CNs).[3] This can be by excluding named entities:

(1) $\forall A.POS(N, A) \wedge \neg SEM\_LABEL(NE, A) \Rightarrow A{:}\textsc{cn}.$

[2] See [12] for a number of arguments as regards the advantages of this move.

[3] This does not mean that we are not interested in instances. To the contrary. What we are saying here is that this rule distinguishes between CNs and instances of these CNs (the difference between a type like Man and an instance of this type, e.g. $John$). There will be a separate rule to derive instances which we do not show here. The type CNis technically a universe, a collection of the names of types into a single type. See [11,3] for more information on the use of the type theoretic notion of universe as this is employed in MTT semantics.

Hyponym and hypernym relations are then defined as subtypes:

(2) $\forall A, B.Hyp(A, B) \Rightarrow A < B$:CN.

(3) $\forall A, B.HyR(A, B) \Rightarrow B < A$:CN.

Synonyms can be defined using equality:[4]

(4) $\forall A, B.Syn(A, B) \Rightarrow A = B$:CN.

Synonymicity is not only relevant for CNs but for other linguistic categories. We can do that as well by changing to:

(5) $\forall A, B.Syn(A, B) \Rightarrow A = B$:$C(C$:$LType)$

$LType$ is a universe of linguistic types, it includes the types instantiated in linguistic semantics (CN, adjectival and verbal types, types for quantifiers etc. See [2] for a discussion).

For instances of terms, like for example proper names, we define the following:

(6) $\forall A, B.Ins(A, B) \Rightarrow A$:$B$

This means that if A is an instance of B then A is of type B. For example, if Einstein is an instance of $Person$, then $Einstein$:$Person$ with $Person$:CN.

2.2 Predicates and world knowledge information

The next question is, how can one extract information on the type of predicates, like for example verbs. JDM provides loads of information with every word, for example characteristics, synonyms, antonyms, collocations. For verbs, *agent*, *patient* and in general thematic relations are defined. These can guide us in assigning types for predicates. In JDM, one can look for semantic relations like $action > agent$, $action > patient$ and various other such relations. For example *man* appears as the agent of a number of verbs that express actions like *question*. There is a further relation, the inverse agent relation, $agent^{-1}$. This relation returns a list of terms (and instances of terms) that can function as the agent for the action denoted by the verb. For example, *question* will involve among others *teacher, mother, child, daugther, person, human*. How can we make sense in order to provide typings in MTTs? Well, there is a straightforward to do this. What we need is to find the most general term, i.e. the term that all the other terms are hyponyms of. Instances of terms are not needed in this process.[5]

[4] Of course, this will treat A and B as perfect synonyms which as we know do not really exist in natural languages. We do not discuss this issue here.

[5] The formula reads as follows: forall A and B, where A is an agent of B (so B is a predicate), if there exists a C such than all A are either hyponyms of C

(7) $\forall A, B.Agent(A, B).\exists C.(Hyp(A, C) \vee (A = C)) \Rightarrow C \rightarrow Prop$

Similar processes can be defined for the patient arguments as well as for adjectives. Adjectives like *grande* can be defined as regards typing by looking at the terms that can have this characteristic. Again, JDM, gives us a list of terms and instances of terms for this reason. For adjectives, we might (depending) on the adjective, give either a typing in the same style as we have proposed above or a polymorphic type extending over a universe which includes the most general type found along with its subtypes:

(8) $\forall A, B.Agent(A, B).\exists C.Hyp(A, C) \Rightarrow C \rightarrow Prop$

(9) $\forall A, B.Char(A, B).\exists C.Hyp(A, C) \Rightarrow \Pi C{:}\mathrm{CN}_C.C \rightarrow Prop$

Due to the abundance of information that JDM has to offer, one can further encode different sorts of information in the form of axioms or definitions. For example the *has_part* relation, in effect a mereological relation, can be translated as a part of relation with $part\_of{:}[\![Object]\!] \rightarrow [\![Object]\!] \rightarrow Prop$.

2.3 Polysemy

The next issue we want to look at is polysemy, most specifically what to do with respect to the translation process in case of polysemous terms. First of all, we have to note here that JDM does not distinguish between homophony and polysemy in the sense they are usualy understood in the literature on formal semantics (e.g. *bank* as homophonous and *book* as polysemous). For JDM, there is only one term to refer to both homophony and polysemy, and this is polysemy. This is what we are going to use here as well, a single notion for all cases where different meanings associated with a given word are found. For JDM, there is this first level where a word with more than one meaning (irrespective of whether the meanings are related or not) are dubbed as polysemous, and then additional levels of refinement where relations between the different meanings can arise (or not). In MTTs, as in formal semantics in general, there are different treatments with respect to cases of homophony and cases of polysemy. For example, in [11], homophony is treated via using local coercions (local subtyping relations) while logical polysemy (cases like *book*) via introducing dot-types, types that encode two senses that do not share any components (see [10] for the formal details). It is a difficult task to be able to translate from a polysemous term identified in JDM

or are equal to C, the predicate $C \rightarrow Prop$ is returned. In case there is no supertype in the refinements, then there are two options: a) introduce a supertype or b) split the refinements into different classes. For example in case we have refinements $human, man, pilot, vehicle, car, bike$, we can split this into class A $= pilot, man < human$ and class $B = bike, car < vehicle$ and propose an overloaded polysemous type for the verb in question, with two different typings, $[\![Human]\!] \rightarrow Prop$ and $[\![Vehicle]\!] \rightarrow Prop$.

to the correct mapping in MTTs. However, there are some preliminary thoughts on how this can be achieved. First of all, let us look at some cases of polysemy identified in JDM that would not be considered such cases in mainstream formal semantics. For example the term *individual* is marked as polysemous in JDM. The reason for this is that JDM goes into more detail than what most formal semantics theories do. JDM distinguishes different meanings of *individual* with respect to its domain of appearance, i.e. the different notion of individual in statistics, biology or administration. This level of fine-grainedness is not found in formal semantics. However, there is no reason why we should not go into this level of detail in MTTs. This can be captured in a translation procedure by presenting the different types according to the different domains. In order to encode domains, we use type theoretic contexts [15,6]. The following translation can be defined for these cases:

(10) $\forall A, B_1, ..., B_n.POS(N, A) \wedge \neg(Instance(PN, A)) \wedge Domain(B_1, ..., B_n) \wedge A(\text{in } B_1, ..., B_n) \Rightarrow A{:}\text{CN in } \Gamma_B 1, A{:}\text{CN in } \Gamma_{B...}, A{:}\text{CN in } \Gamma_B n$

What about other cases of polysemy like for example *book* or *bank*? One way to look at the translation process in these cases is the following: In case a term is dubbed polysemous in JDM, we look at the semantic refinements and introduce all these refinements as subtypes of the initial term:

(11) $\forall A, B_1, ..., B_n.POS(N, A) \wedge \neg(POS(PN, A)) \wedge Ref(A, (B_1, ..., B_n)) \Rightarrow A < B_1, ..., B_n{:}\text{CN}$

Now in order to decide whether we are going to use local coercions or dot-types we proceed as follows: the types that participate in dot-types are limited and enumerable: some of these include $[\![Phy]\!]$, $[\![Info]\!]$, $[\![Event]\!]$, $[\![Inst]\!]$ among others. We can thus create such a set of refinements that can be senses of a dot-type. Call this set *dot refinements*, DR. Now, in case the refinements happen to be members of this set then we can form a dot-type out of the individual refinements:

(12) $\forall A, B_1...B_n.POS(N, A) \wedge Ref(A(B_1, ..., B_n))A, B \in DR \Rightarrow A{:}CN < B_1 \bullet B... \bullet B_n$

More information on ways to approach the translation procedure with respect to polysemy will be given in the full paper. There, other cases will be discussed: a) cases where the two meanings are associated with different types (e.g. a case where one meaning is verbal and the other adjectival), b) Cases where a combination of approaches might be needed (e.g. a polysemous noun that further has adjectival or verbal meanings). Further issues of how to use world-knowledge information drawn from JDM and in which way will also be discussed. Last but not least, we are going to discuss how can such an idea be implemented in a system that will take as input information from JDM and will output MTT representations (to be used for example in the proof-assistant Coq).

References

1. C. Benzmüller, Theiss F., and Fietzke A. The LEO-II project. In *Automated Reasoning Workshop*, 2007.
2. S. Chatzikyriakidis and Z. Luo. An account of natural language coordination in type theory with coercive subtyping. In Y. Parmentier and D. Duchier, editors, *Proc. of Constraint Solving and Language Processing (CSLP12). LNCS 8114*, pages 31–51, Orleans, 2012.
3. S. Chatzikyriakidis and Z. Luo. Adjectives in a modern type-theoretical setting. In G. Morrill and J.M Nederhof, editors, *Proceedings of Formal Grammar 2013. LNCS 8036*, pages 159–174, 2013.
4. S. Chatzikyriakidis and Z. Luo. Natural language inference in Coq. *J. of Logic, Language and Information.*, 23(4):441–480, 2014.
5. S. Chatzikyriakidis and Z. Luo. Natural language reasoning using proof-assistant technology: Rich typing and beyond. In *Proceedings of EACL2014*, 2014.
6. S. Chatzikyriakidis and Z. Luo. Using signatures in type theory to represent situations. *Logic and Engineering of Natural Language Semantics 11. Tokyo*, 2014.
7. Robin Cooper, Simon Dobnik, Shalom Lappin, and Staffan Larsson. A probabilistic rich type theory for semantic interpretation. In *Proceedings of the European Associaton of Computational Linguistics*, 2014.
8. C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT press, 1998.
9. M. Lafourcade. Making people play for lexical acquisition with the jeuxdemots prototype. In *SNLP'07: 7th international symposium on natural language processing*, page 7, 2007.
10. Z. Luo. Type-theoretical semantics with coercive subtyping. *Semantics and Linguistic Theory 20 (SALT20), Vancouver*, 2010.
11. Z. Luo. Contextual analysis of word meanings in type-theoretical semantics. *Logical Aspects of Computational Linguistics (LACL'2011). LNAI 6736*, 2011.
12. Z. Luo. Common nouns as types. In D. Bechet and A. Dikovsky, editors, *Logical Aspects of Computational Linguistics (LACL'2012). LNCS 7351*, 2012.
13. P. Martin-Löf. An intuitionistic theory of types: predicative part. In H.Rose and J.C.Shepherdson, editors, *Logic Colloquium'73*, 1975.
14. P. Martin-Löf. *Intuitionistic Type Theory*. Bibliopolis, 1984.
15. A. Ranta. *Type-Theoretical Grammar*. Oxford University Press, 1994.
16. C. Retoré. The montagovian generative lexicon Tyn: a type theoretical framework for natural language semantics. In R. Matthes and A. Schubert, editors, *Proc of TYPES2013*, 2013.

Perceptual Meaning in TTR Judgement-based Semantics and Conceptual Spaces

Staffan Larsson

Dept. of Philosophy, Linguistics and Theory of Science
University of Gothenburg, Sweden

Abstract. We are developing a type-theoretical judgement-based semantics where notions such as perception, classification, judgement, learning and dialogue coordination play a central role. By bringing perception and semantic coordination into formal semantics, this theory can be seen as an attempt at unifying cognitive and formal approaches to meaning. The purpouse of this paper is to briefly compare judgement-based semantics to the theory of conceptual spaces. We argue that the former enables integration of perceptual aspects of meaning with those traditionally studied in formal semantics, and furthermore that it enables computational modeling and implementation of these aspects of meaning.

1 Introduction

How is linguistic meaning related to perception and the world? How do words acquire their meaning? These are two central questions for any theory of meaning in natural language (Miller and Johnson-Laird, 1976) . We are developing a type-theoretical *judgement-based semantics* where notions such as perception, classification, judgement, learning and dialogue coordination play a central role (Cooper, 2005; Cooper and Larsson, 2009; Larsson, 2011; Dobnik *et al.*, 2011; Cooper, 2012; Dobnik and Cooper, 2013; Cooper *et al.*, 2015; Larsson, 2013). Meaning is regarded as being acquired by an agent through its perception of, and interaction with, the world and other agents. This makes meaning agent-relative but essentially social (in the sense of being coordinated in interaction between individuals) and dynamic (in the sense of always being up for revision and negotiation as new perceptual and conversationally mediated information is encountered). By bringing perception and semantic coordination into formal semantics, this theory can be seen as an attempt at unifying cognitive and formal approaches to meaning. The purpouse of this paper is to briefly compare judgement-based semantics to the theory of conceptual spaces (Gärdenfors, 2000; Gärdenfors, 2004).

2 Judgement-based semantics and perceptual meaning

We will here take for granted that relating meaning to perception is of central importance when accounting for the meaning of linguistic expressions which refer to the physical world; for a motivation see Larsson (2013). Knowing the meaning of an expression is related to an agent's ability to identify perceived objects and situations which can be referred to by the expression. For example, knowing the meaning of "blue" is intimately connected with an agent's ability to correctly identify blue objects. Similarly, an agent's ability to assign a meaning to "a boy hugs a dog" is related to her ability to correctly classify perceived situations where a boy hugs a dog. An important difference to traditional possible worlds semantics (Montague, 1974) is that we focus on modelling perceptual mechanisms which start from perceptual raw data, rather than giving an abstract representation of them as functions from possible worlds and times to objects in a space of denotations as in the Montagovian approach. This allows us to provide a more concrete account of the nature of the link between language and the world.

To make the link between "low-level" perceptual data and "high-level" formal linguistic representations, we will use the notion of a (statistical) classifier, a computational device determining what class an item belongs to, based on various properties of the item. Crucially, the information fed to a classifier need not be encoded in some high-level representation language (such as logic or natural language). Instead, it may consist entirely of empirical raw data encoding "low-level" information about the item in question. The idea of using classifiers (or more specifically, connectionist models) to represent meanings was first put forward by Harnad (1990) as a way of addressing the "symbol grounding problem" in artificial intelligence, and is consistent with several theories of word meaning as grounded in sensory (or *embodied*) representations which have emerged during the last decade or so (Roy, 2005; Steels and Belpaeme, 2005).

To integrate classification of perceptual data with formal semantics, we are using TTR (Type Theory with Records), a framework developed with a view to giving an abstract formal account of natural language interpretation (Cooper, 2012), as our formalism and foundational semantic theory. TTR starts from the idea that information and meaning is founded on our ability to perceive and classify the world, i.e., to perceive objects and situations as being of types. In TTR, types are first-class objects, which allows perceptual classifier functions to be formalised and used in representing meanings of linguistic expressions together with the high-level conceptual aspects of meaning traditionally studied in formal semantics. Semantic phenomena which have been described using TTR include modelling of intensionality and mental attitudes (Cooper, 2005), dynamic generalised quantifiers (Cooper, 2004), co-predication and dot types in lexical innovation, frame semantics for temporal reasoning, reasoning in hypothetical contexts (Cooper, 2011), enthymematic reasoning (Breitholtz and Cooper, 2011), clarification requests (Cooper, 2010), negation (Cooper and Ginzburg, 2011), and information states in dialogue (Cooper, 1998; Ginzburg, 2012).

Larsson (2011) and Larsson (2013) show how a simple classifier of sensory information based on the perceptron can be cast in TTR, and how an agent can learn from interaction by training the classifier based on linguistic and perceptual input. Linguistic input, e.g. utterance of "That's red" (assuming a situation where two agents are inspecting a colour sample), is interpreted as a function from a situation where some object is in the (shared) focus of attention and there is a sensor reading (e.g. in the form of a real-numbered vector, but shown below as a colour patch) from a colour sensor, to a judgement that the situation is of a type where the object is judged to be red.

$$\begin{bmatrix} \text{foc-obj} & = \text{a} \\ \text{sensor}_{colour} & = \blacksquare \end{bmatrix} : \text{red(a)}$$

To achieve this, the function contains a classifier which takes the (real-numbered vector corresponding to the perception of the) colour sample and produces a judgement whether the vector is within the borders of redness. For details on how classifiers are embedded into TTR functions, see Larsson (2013), which also includes a brief account of compositionality.

Categorical judgments of the kind exemplified above are of course not suited for accounting for vagueness and other gradient semantic phenomena. To remedy this, a probabilistic extension of TTR has been developed (Cooper *et al.*, 2014, 2015). For an account of vagueness in perception using probabilistic TTR, where judgements are associated with probabilities, see Fernández and Larsson (2014). Below is an example of a probabilistic judgement where a situation is judged, with a probability of 0.79, to be one where the object in the focus of attention is red.

$$\begin{bmatrix} \text{foc-obj} & = \text{a} \\ \text{sensor}_{colour} & = \blacksquare \end{bmatrix} :_{0.79} \text{red(a)}$$

In probabilistic TTR, the result of an act of classification is represented as a probability distribution over type assignments, i.e., as a set of probabilistic judgements where the probabilities sum to 1.

3 Observations in conceptual spaces

There are interesting connections between the idea of using classifiers to model perceptual meaning, and the notion of conceptual spaces. Gärdenfors distinguishes three levels of modeling concepts and reasoning: symbolic, subconceptual, and conceptual.

Reasoning on the *symbolic level* is framed as operations on propositions expressed by symbolic structures (i.e., symbol manipulation according to explicit rules), and focuses on computing logical consequences (i.e. deductive reasoning). Gärdenfors also discusses connectionist models of meaning as an example of representations on a *subconceptual* level, where reasoning modelled by the activities of the artificial neurons. Concepts on the subconceptual are modelled

"implicitly", in contrast to the more explicit *conceptual* level where concepts are modelled as geometrical structures (points, vectors and regions) in conceptual spaces. Conceptual reasoning is described in terms of distances in a space, and focuses on modeling reasoning about concepts, in particular inductive and nonmonotonic reasoning.

Corresponding to these three kinds of reasoning, Gärdenfors makes a distinction between three ways of describing an observation, which we will use to frame our discussion. On the symbolic level, observations are described in some specified language with a fixed set of primitive predicates. Denotations of predicates assumed to be known, and observational statements furnished to reasoner by incorrigible perceptual mechanisms. On the conceptual level, concepts characterised in terms of some underlying conceptual space, consisting of a number of "quality dimensions" (or *domains*). An observation on this level is an assignment to an object of a location in a conceptual space. For example, an observation that "x is red" is expressed by assigning x a point in colour space. Finally, on the subconceptual level an observation is regarded as something which is received by our sensory organs, or in general some kind of *receptors*, including e.g. measuring instruments.

4 Perceptual judgements and conceptual spaces

Below, we will explore the relation between conceptual spaces and classifiers as different (but to some extent complementary) ways of capturing perceptual meaning. Roughly, the correspondence is the following: classification events can be regarded as making a judgement as to whether an observation falls within that region in a conceptual space. Also, classifier learning can be regarding as defining areas (regions or volumes) within a conceptual space.

4.1 Sensor readings and the subsymbolic level

It appears fairly obvious that the sensor readings in TTR correspond to Gärdenfors' subsymbolic representations. Both represent low-level perceptual input using vectors, points or regions in vector spaces. In judgement-based semantics, sensor readings are the input to classifiers.

4.2 Types and the symbolic level

Gärdenfors' symbolic representations of observations appears to correspond to the types which result from judgements. However, in TTR we do not assume that there is a fixed set of primitive predicates. Instead, we are interested in modeling concept learning (Larsson, 2013). Perhaps even more importantly, we do not assume that the extensions (denotations) of predicates are known. Instead, we represent meanings of concrete expressions using classifiers which take some perceptual input and produce a judgement. The classifiers can be thought of as representing the *intensions* of linguistic expressions. Since these classifiers

can be trained, they are also dynamic and learnable. Furthermore, TTR does not assume that perceptual mechanisms are always correct, nor that agents always agree on their perceptions of a situation or their judgements about the situation. Finally, a nice feature of TTR is not only properties but also relations are types which opens the way for compositional semantics involving predicates of arbitrary arity.

4.3 Classifiers, judgements and the conceptual level

According to Gärdenfors, an observation on the conceptual level is an assignment to an object of a location in a space. In TTR, this corresponds to an act of classification producing a (possibly probabilistic or graded) judgement concerning (the probability of) a situation being of a certain type, thus mediating between the sensor reading and the high-level "symbolic" types. In this way, classifiers connects subsymbolic observations and semantic concepts to "symbolic" reasoning.

Induction is indeed closely related to concept formation, since our concepts are formed and learned by induction from observations (including observations in interaction). Gärdenfors' claim that "induction can be seen as establishing connections between various kinds of input" is echoed in TTR in that classifiers are trained by generalising over several instances, thereby connecting several instances. An advantage of classifiers is that they are straightforwardly implementable, and that classification on sensory input is a well-studied research area.

5 Conclusion

We have compared judgement-based semantics with conceptual spaces, and concluded that there are important similarities but also some differences. One aim of TTR judgement-based semantics is to formalise semantic classification and learning in detail, to enable integration of these aspects of meaning with those traditionally studied in formal semantics, and to enable computational modeling and implementation of these aspects of meaning. By using statistical classifiers, we connect to machine learning theory, giving access to a host of classification methods and associated learning algorithms.

Bibliography

Breitholtz, E. and Cooper, R. (2011). Enthymemes as rhetorical resources. In *Proceedings of the 15th Workshop on the Semantics and Pragmatics of Dialogue (SemDial 2011)*, pages 149–157, Los Angeles (USA).

Cooper, R. (1998). Information states, attitudes and dependent record types. In *ITALLC98*, pages 85–106.

Cooper, R. (2004). Dynamic generalised quantifiers and hypothetical contexts. In *Ursus Philosophicus, a festschrift for Björn Haglund*. Department of Philosophy, University of Gothenburg.

Cooper, R. (2005). Austinian truth, attitudes and type theory. *Research on Language and Computation*, **3**, 333–362.

Cooper, R. (2010). Generalized quantifiers and clarification content. In P. Łupkowski and M. Purver, editors, *Aspects of Semantics and Pragmatics of Dialogue. SemDial 2010, 14th Workshop on the Semantics and Pragmatics of Dialogue*, Poznań. Polish Society for Cognitive Science.

Cooper, R. (2011). Copredication, quantification and frames. In S. Pogodalla and J.-P. Prost, editors, *LACL*, volume 6736 of *Lecture Notes in Computer Science*, pages 64–79. Springer.

Cooper, R. (2012). Type theory and semantics in flux. In R. Kempson, N. Asher, and T. Fernando, editors, *Handbook of the Philosophy of Science*, volume 14: Philosophy of Linguistics. Elsevier BV. General editors: Dov M. Gabbay, Paul Thagard and John Woods.

Cooper, R. and Ginzburg, J. (2011). Negation in dialogue. In *Proceedings of the 15th Workshop on the Semantics and Pragmatics of Dialogue (SemDial 2011)*, Los Angeles (USA).

Cooper, R. and Larsson, S. (2009). Compositional and ontological semantics in learning from corrective feedback and explicit definition. In J. Edlund, J. Gustafson, A. Hjalmarsson, and G. Skantze, editors, *Proceedings of DiaHolmia, 2009 Workshop on the Semantics and Pragmatics of Dialogue*.

Cooper, R., Dobnik, S., Lappin, S., and Larsson, S. (2014). A probabilistic rich type theory for semantic interpretation. In *Proceedings of the EACL Workshop on Type Theory and Natural Language Semantics (TTNLS)*.

Cooper, R., Dobnik, S., Lappin, S., and Staffan, L. (2015). Probabilistic type theory and natural language semantics. Under review.

Dobnik, S. and Cooper, R. (2013). Spatial descriptions in type theory with records. In *Proceedings of IWCS 2013 Workshop on Computational Models of Spatial Language Interpretation and Generation (CoSLI-3)*, pages 1–6, Potsdam, Germany. Association for Computational Linguistics.

Dobnik, S., Larsson, S., and Cooper, R. (2011). Toward perceptually grounded formal semantics. In *Proceedings of the Workshop on Integrating Language and Vision at NIPS 2011*, Sierra Nevada, Spain. Neural Information Processing Systems Foundation (NIPS).

Fernández, R. and Larsson, S. (2014). Vagueness and learning: A type-theoretic approach. In *Proceedings of the 3rd Joint Conference on Lexical and Computational Semantics (∗SEM 2014)*.

Gärdenfors, P. (2000). *Conceptual spaces - the geometry of thought*. MIT Press.

Gärdenfors, P. (2004). Conceptual spaces as a framework for knowledge representation. *Mind and Matter*, **2**(2), 9–27.

Ginzburg, J. (2012). *The Interactive Stance*. Oxford University Press, New York.

Harnad, S. (1990). The symbol grounding problem. *Physica D: Nonlinear Phenomena*, **42**(1990), 335–346.

Larsson, S. (2011). The ttr perceptron: Dynamic perceptual meanings and semantic coordination. In *Proceedings of the 15th Workshop on the Semantics and Pragmatics of Dialogue (SemDial 2011)*, Los Angeles (USA).

Larsson, S. (2013). Formal semantics for perceptual classification. *Journal of Logic and Computation*.

Miller, G. A. and Johnson-Laird, P. N. (1976). *Language and perception*. Belknap Press.

Montague, R. (1974). *Formal Philosophy: Selected Papers of Richard Montague*. Yale University Press, New Haven. ed. and with an introduction by Richmond H. Thomason.

Roy, D. (2005). Grounding words in perception and action: computational insights. *Trends in Cognitive Sciences*, **9**(8), 389–396.

Steels, L. and Belpaeme, T. (2005). Coordinating perceptually grounded categories through language: A case study for colour. *Behavioral and Brain Sciences*, **28**(4), 469–89. Target Paper, discussion 489-529.

Interfacing Language, Spatial Perception and Cognition in Type Theory with Records

Simon Dobnik*

Dept. of Philosophy, Linguistics & Theory of Science
Centre for Language Technology
University of Gothenburg, Sweden
simon.dobnik@gu.se
http://flov.gu.se

Abstract. We argue that computational modelling of perception, action, language, and cognition introduces several requirements on a formal semantic theory and its practical implementations. Using examples of semantic representations of spatial descriptions we show how Type Theory with Records (TTR) satisfies these requirements. The advantage of truth being based on agent-relative judgements in TTR is crucial in this but practically it comes with a computational cost. We argue that the number of type judgements an agent has to make can be minimised by incorporating a cognitive notion of judgement that is driven by perceptual attention.

Keywords: spatial language, Type Theory with Records (TTR), attention driven judgements, computational framework

In the proposed presentation we overview and connect two lines of our work related to Type Theory with Records (TTR) [2, 3]: modelling of spatial language and cognition [8] and modelling attention-driven judgement [18].

Cross-disciplinary research has shown that spatial language is dependent on several contextual factors that are part of an agent's interaction with the environment through perception and other agents through dialogue, for example geometrical arrangement of the scene [28], the type of objects referred to and their interaction [5], visual and discourse salience of objects [17], alignment in dialogue [32], and gesture [31] among others. Although the contribution of these contextual factors has been well studied in psychology, computational linguistics, computer science, geo-information science and robotics, several questions relating to representing their semantics and building formal computational models for situated agents still remain. These relate to (i) how an agent is able to determine *the sense and reference* of spatial descriptions; (ii) *grounding and information fusion* of contextual features into bundles of meaning representations; (iii) *bridging of perceptual and conceptual domains*; (iv) *formal accuracy and*

* I am grateful to Robin Cooper, Staffan Larsson and John D. Kelleher for discussion which has lead to significant changes in this paper.

sufficient expressiveness of representations for modelling human reasoning; (v) their *compositionality* with meaning representations of other words, sentences and utterances; (vi) their *adaptability and learnability* by an agent in new physical and conversational contexts.

In building situated conversational agents, several systems have been proposed but none of them capture all of these requirements. For example, *semiotic schemas* [29] account for the meaning of words that define perceivable entities and performable actions but it is not straightforwardly evident how they relate to other linguistic representations. [20] adopt a layered model with distinct representations at each layer. Although there exist mechanisms by which these representational levels interact, the kinds of representations at each level are quite distinct from each other and are shaped by different operations. The question we would like to address is whether such representational levels and operations can be generalised by taking inspiration from the way humans assign, learn and reason with meaning.

Classical formal semantics based on first order logic [11, 1] provides the required formal accuracy and expressiveness of a representation system. However, it mainly on explaining how meaning representations of words are composed to form meaning representations of sentences and does not address how meaning (both *sense* or *intension* and *reference* or *extension*) is learned and assigned in perception. The analyses of spatial descriptions are represented in first order logic such as: on(x,y)_1: object(x) $\wedge$ object(y) $\wedge$ supports(y,x) $\wedge$ contiguous(surface(x),surface(y)) and on(x,y)_2: object(x) $\wedge$ object(y) $\wedge$ contiguous(boundary(x),y), see for example [26, 15]. The analysis tell us that the meaning of spatial descriptions is composed of several geometric primitives (surface/1, contiguous/2, boundary/1) but the meaning of these primitives is left un-accounted for. In model-theoretic semantics the expression's reference is determined by an assignment in a form of a valuation function between the linguistics strings and entities (or sets of tuples of entities) in a model. The model is agent external and fixed. The valuation returns true if an entity or a relation between entities denoted by an expression can be found in the model, otherwise it returns false. While it would be possible to represent the referential semantics of a "on" in a model by listing a set of all the coordinates of the locations where this spatial description applies, this referential representation of meaning is cumbersome as the model would have to represent for every scale, for every spatial relation, for every pair of objects. Note also that angles and distances in a coordinate system are continuous measures which means that such sets would be infinite. Furthermore, the way humans refer to space is vague (an object may be "near" another object depending on several contextual factors) and there is gradience of reference (some objects are "nearer" the landmark than others. Both vagueness and gradience of spatial language is captured in computational models as spatial templates or potential fields. While spatial templates can be thought of as referential overlays of regions induced experimentally (as a set of points where participants consider a particular spatial relation to apply) [25], potential fields capture the notion that such regions can be generalised as func-

tions [13, 28]. However, these functions do not represent objects in the model (or the extension or referential meaning of these descriptions) but rather capture their *sense* or *intension*: in what ways a description relates to perceptual observations. Knowing this function we can check whether a particular spatial relation associated with the function applies for particular pair of objects and to what degree. In addition to angle and distance, several contextual parameters can be incorporated, for example the presence of distractor objects [4], object occlusion [19], etc. or the function itself can be learned from the dataset of perceptual observations and descriptions as a classifier [30, 7]. The notion of applying a function representing the meaning of words to the perceptual observations is also known as *grounding* these words in perception [14]. The grounded meanings of spatial descriptions or their senses can be thought of as bundles of several distinct yet interacting sources of information organised at different levels of conceptual abstraction, ranging from sub-conceptual perceptual information to contents of entire dialogue interactions in an information state of an agent [22].

Model-theoretic approach to semantics assumes that the model is given (derived through some external process), complete and represents a state of affairs at a particular temporal snapshot [12]. However, practically complete models may be rarely observable and we must deal with partial models. We must also account for the fact that the we may incrementally observe more and more of the world and we have to update the model with new observations, sometimes even correct the representation that we have already built in light of the new evidence. Finally, the world is not static itself as new new objects and events continuously come into existence. Imagine a robot (and indeed such robots were used in the early days of robotics) with a pre-programmed static model of the world. Every minute change in the world would render it useless as there would be a discrepancy between its representation of the world and the actual world. Modern robotic models used in localisation and map building are incrementally learned or updated over time by taking into account robot's perception and motion and errors associated with both [6]. An important consequence of this is that the model of the world a robot builds is individual to a particular robot's life-span and experience. Two robots experiencing the same world will have a slightly different models. Of course, the more they experience the world, the more similar the models will be. It is conceivable that humans learn meanings in the same way. However, doing so they are equipped with yet another tool to overcome individual inconsistencies in their model. They can use linguistic dialogue interaction to resolve such inconsistencies in the form of repair [27].

Type Theory with Records (TTR) builds on the tradition of the classical formal semantics (and therefore captures the notion of compositionality) but at the same time, drawing on insights from situation semantics, addresses the outstanding questions related to perception discussed in the preceding paragraphs. It starts from the idea that information is founded on our ability to perceive and classify the world, that is to perceive or *judge* objects and situations as being of types. Types are intensional - that is, there can be distinct types which have

identical extensions. In this way sense is derived operationally as a computable function [21] or a classifier [23]. The notion of truth is linked to judgements that an object a is of type T ($a : T$). Under this view the type inventory is internal to an agent as types are learned and continuously refined by each agent as it encounters new situations [23]. Agents converge on sufficiently similar type representations which are a requirement for successful communication because they are part of the same perceptual and discourse contexts that impose external constraints on it, for example in terms of corrective feedback or the differences of what an agent expects to perceive and what it perceives. In such a situated dialogue learning scenario the TTR system can be also given a Bayesian probabilistic interpretation [3]. Because TTR relates perception directly to higher-level conceptual reasoning in a probabilistic way which allows modelling of gradience which makes it suitable for modelling semantics of spatial descriptions.

In contrast to the classical model-theoretic framework where types are used for the purpose of compositionality (the denotations of phrases are either model objects of basic types such as entities and truth values or functions composed from these types), TTR introduces an extended set of basic types (for example *Ind* and *Real* that correspond to basic human conceptual categories such as individuals and real numbers) and *a rich type system* which contains arbitrarily complex record types which are able to, among other things, express complex lexical semantics and dialogue information states. The proof objects of record types are records. Records and record types are similar to feature structures containing label-value pairs. The information expressed in types can be compared and reasoned about as the type systems allows *subtype* and *dependent type* relations. The ability of TTR to represent hierarchically organised multi-source information fulfils another requirement for modelling spatial descriptions.

In this presentation we discuss how our empirical investigations of learning geometric meanings of spatial descriptions with situated robots [7], learning functional meanings of prepositions from collections of image descriptions [9], and modelling of reference frame assignment in conversation [10] can be formulated in the TTR framework. For examples and details of TTR formulations refer to [8]. The overall goal is to provide an account of semantics of spatial prepositions for these modalities, and over the longer term, use the framework as a knowledge representation system of a situated agent.

This leads to a question how well as a semantic framework TTR is practically suited as a semantic representation layer for embodied agents. Humans are very flexible in assigning meaning and naturally we would like to preserve the same flexibility in our framework. New types can be created or learned by an agent dynamically. Furthermore, the record types allow us to construct the following relations between types which allow us to compare and reasoning about meaning:

– Intensionality/non-exclusivity of types: an object may belong to more than one type which may be structurally (nearly) an entirely different representation. For example, a sensory reading of a particular situation in the world involving spatial arrangement of objects may be assigned several record types

of spatial relations simultaneously, each with a unique internal structure: *Left*, *Near*, *At*, *Behind*, etc.
- A type may be a subtype of another type. An object judged as being of a particular type is also of all types that this type is a sub-type of: given that *Chair* is a sub-type of *Object*, a situation of type *Chair* is also of type *Object*.
- A type may be a component of another type. An object of the first type is partially matched with the second type. For example, a situation of type *Chair* is a component of the situation of *Table-Left-Chair*.
- A type may be a dependent type of another type. For example, the type *Left* is a dependent type of *Table-Left-Chair*. In order to judge a situation to be of type *Table-Left-Chair* one has to judge it to be of type *Left*.

Since each type assignment involves a binary judgement (something is of a type T or not) for each record of situation an agent having an inventory of n types can make n assignments. Learning what types a particular situation can be assigned involves 2^n possible outcomes, hence for $n = 3$, $2^3 = 8$: $\{\}$, $\{T_1\}$, $\{T_2\}$, $\{T_3\}$, $\{T_1, T_2\}$, $\{T_1, T_3\}$, $\{T_2, T_3\}$ and $\{T_1, T_2, T_3\}$, but if types are sub-types or dependent of another the number of judgements could be reduced.

We argue that agents such as situated robots need (i) a judgement control mechanism and (ii) a method for organising their type inventory [18]. For (i) we propose the Load Theory of selective attention and cognitive control [24] to be a suitable candidate. This model of attention distinguishes between two mechanisms of selective attention: *perceptual selection* and *cognitive control*. Following this theory, we argue that type judgements can be grouped into three categories: (i) pre-attentive, (ii) task induced, and (iii) context induced judgements. An agent makes the first kind of judgements continuously, but varying strategies depending on its cognitive load. The other two kinds of judgements are primed by the task and the physical context that the agent is engaged with. We propose that agents organise their inventory of types that fall under (ii) and (iii) into subsets or bundles (computationally they can be modelled as lists) that are associated with the agent's cognitive states. The states can be modelled as Partially Observable Markov Decision Processes (POMDPs, [16]) and can be thought of as sensitivities towards certain objects, events, and situations. The states of a POMDP network are connected by actions, in this case the priming policies (ii) and (iii). The types an agent actually perceives in each state represent the observations for each state (note that here we are not dealing with learning new types). The model ensures that observing certain types at a particular state primes the agent to observe particular other types in the states following it. Hence, past experience primes the agent to observe new situations. The reward function is governed by the benefit of an agent being primed to perceive the world this way.

In this paper we outlined an application of type theory to natural language semantics and demonstrated how the TTR framework allows us to relate the semantics to action, perception and cognition. Furthermore, on the example of spatial descriptions we argued tat natural language, perception and cognition put high demands on the expressiveness of the type theoretic framework which

is associated with high computational cost. In order to counter this, we proposed a method to limit the possible type judgements of an agent by separate cognitive attentual mechanism which we will be testing in practical implementations with situated agents in our forthcoming work.

References

1. Blackburn, P., Bos, J.: Representation and inference for natural language. A first course in computational semantics. CSLI Publications (2005)
2. Cooper, R.: Type theory and semantics in flux. In: Kempson, R., Asher, N., Fernando, T. (eds.) Handbook of the Philosophy of Science, General editors: Dov M Gabbay, Paul Thagard and John Woods, vol. 14. Elsevier BV (2012)
3. Cooper, R., Dobnik, S., Lappin, S., Larsson, S.: A probabilistic rich type theory for semantic interpretation. In: Cooper, R., Dobnik, S., Lappin, S., Larsson, S. (eds.) Proceedings of the EACL 2014 Workshop on Type Theory and Natural Language Semantics (TTNLS). pp. 72–79. Association for Computational Linguistics, Gothenburg, Sweden (27 April 2014)
4. Costello, F.J., Kelleher, J.D.: Spatial prepositions in context: the semantics of near in the presence of distractor objects. In: Proceedings of the Third ACL-SIGSEM Workshop on Prepositions. pp. 1–8. Prepositions '06, Association for Computational Linguistics, Stroudsburg, PA, USA (2006)
5. Coventry, K.R., Prat-Sala, M., Richards, L.: The interplay between geometry and function in the apprehension of Over, Under, Above and Below. Journal of Memory and Language 44(3), 376–398 (2001)
6. Dissanayake, M.W.M.G., Newman, P.M., Durrant-Whyte, H.F., Clark, S., Csorba, M.: A solution to the simultaneous localization and map building (SLAM) problem. IEEE Transactions on Robotic and Automation 17(3), 229–241 (2001)
7. Dobnik, S.: Teaching mobile robots to use spatial words. Ph.D. thesis, University of Oxford: Faculty of Linguistics, Philology and Phonetics and The Queen's College, Oxford, United Kingdom (September 4 2009)
8. Dobnik, S., Cooper, R., Larsson, S.: Type Theory with Records: a general framework for modelling spatial language. In: Dobnik, S., Cooper, R., Larsson, S. (eds.) Proceedings of The Second Workshop on Action, Perception and Language (APL'2). The Fifth Swedish Language Technology Conference (SLTC), Uppsala, Sweden (13 November 2014)
9. Dobnik, S., Kelleher, J.: Exploration of functional semantics of prepositions from corpora of descriptions of visual scenes. In: Proceedings of the Third V&L Net Workshop on Vision and Language. pp. 33–37. Dublin City University and the Association for Computational Linguistics, Dublin, Ireland (August 2014)
10. Dobnik, S., Kelleher, J.D., Koniaris, C.: Priming and alignment of frame of reference in situated conversation. In: Rieser, V., Muller, P. (eds.) Proceedings of DialWatt - Semdial 2014: The 18th Workshop on the Semantics and Pragmatics of Dialogue. pp. 43–52. Edinburgh (1–3 September 2014)
11. Dowty, D.R., Wall, R.E., Peters, S.: Introduction to Montague semantics. D. Reidel Pub. Co., Dordrecht, Holland (1981)
12. Fagin, R., Halpern, J.Y., Moses, Y., Y. Vardi, M.: Reasoning about knowledge. MIT Press, Cambridge, Mass. (1995)
13. Gapp, K.P.: Basic meanings of spatial relations: Computation and evaluation in 3d space. In: Hayes-Roth, B., Korf, R.E. (eds.) AAAI. pp. 1393–1398. AAAI Press/The MIT Press (1994)

14. Harnad, S.: The symbol grounding problem. Physica D 42(1–3), 335–346 (June 1990)
15. Herskovits, A.: Language and spatial cognition: an interdisciplinary study of the prepositions in English. Cambridge University Press, Cambridge (1986)
16. Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. Artificial intelligence 101(1), 99–134 (1998)
17. Kelleher, J., Costello, F., van Genabith, J.: Dynamically structuring updating and interrelating representations of visual and linguistic discourse. Artificial Intelligence 167, 62–102 (2005)
18. Kelleher, J.D., Dobnik, S.: A model for attention-driven judgements in Type Theory with Records. In: Kempson, R., Purver, M. (eds.) Proceedings of the Worshop on Interactive Meaning Construction at the International Workshop on Computational Semantics (IWCS 2015). pp. 13–14. Queen Mary University of London (14 April 2015)
19. Kelleher, J.D., Ross, R.J., Sloan, C., Namee, B.: The effect of occlusion on the semantics of projective spatial terms: a case study in grounding language in perception. Cognitive Processing 12(1), 95–108 (2011)
20. Kruijff, G.J.M., Zender, H., Jensfelt, P., Christensen, H.I.: Situated dialogue and spatial organization: what, where... and why? International Journal of Advanced Robotic Systems 4(1), 125–138 (2007), special issue on human and robot interactive communication
21. Lappin, S.: Intensions as computable functions. Linguistic Issues in Language Technology 9, 1–12 (2013)
22. Larsson, S.: Issue-based Dialogue Management. Ph.D. thesis, University of Gothenburg. (2002)
23. Larsson, S.: Formal semantics for perceptual classification. Journal of Logic and Computation online, 1–35 (December 18 2013)
24. Lavie, N., Hirst, A., de Fockert, J.W., Viding, E.: Load theory of selective attention and cognitive control. Journal of Experimental Psychology: General 133(3), 339–354 (2004)
25. Logan, G.D., Sadler, D.D.: A computational analysis of the apprehension of spatial relations. In: Bloom, P., Peterson, M.A., Nadel, L., Garrett, M.F. (eds.) Language and Space, pp. 493–530. MIT Press, Cambridge, MA (1996)
26. Miller, G.A., Johnson-Laird, P.N.: Language and perception. Cambridge University Press, Cambridge (1976)
27. Pickering, M.J., Garrod, S.: Toward a mechanistic psychology of dialogue. Behavioral and Brain Sciences 27(2), 169–190 (2004)
28. Regier, T., Carlson, L.A.: Grounding spatial language in perception: an empirical and computational investigation. Journal of Experimental Psychology: General 130(2), 273–298 (2001)
29. Roy, D.: Semiotic schemas: a framework for grounding language in action and perception. Artificial Intelligence 167(1-2), 170–205 (Sep 2005)
30. Roy, D.K.: Learning visually-grounded words and syntax for a scene description task. Computer speech and language 16(3), 353–385 (2002)
31. Tutton, M.: A new approach to analysing static locative expressions. Language and Cognition 5, 25–60 (3 2013)
32. Watson, M.E., Pickering, M.J., Branigan, H.P.: Alignment of reference frames in dialogue. In: Proceedings of the 26th Annual Conference of the Cognitive Science Society. Chicago, USA (2004)

Probabilistic Mereological TTR and the Mass/Count Distinction

Peter Sutton and Hana Filip

Heinrich-Heine-Universität Düsseldorf, 40225 Düsseldorf, Germany,
peter.r.sutton.@icloud.com; hana.filip@gmail.com

Abstract. We propose that countability of nouns is not a matter of just a bipartite mass vs. count distinction. Instead we distinguish four classes of nouns. We propose that these divisions can be modelled in a mereological enrichment of prob-TTR (Cooper et al. 2014), via patterns that emerge in probabilistic type judgements associated with them.

Keywords: lattice theoretic semantics, Type Theory with Records, Bayesian Semantics

1 Introduction

The goals of this paper are threefold. (1) We argue that four semantic classes can be used to predict cross and intralinguistic variation in whether nouns are encoded as count or mass. These semantic classes are defined in terms of two kinds of sources: vagueness and overlap. (2) We enrich prob-TTR (Cooper et al. 2014, a probabilistic variant of TTR, Cooper 2012) with mereological relations to give probM-TTR. We adopt prob-TTR since its probabilistic basis is perfectly suited to the representation of vagueness. We enrich it so as to be able to express when two (mereological) entities overlap. (3) We derive our four classes by showing how vagueness and overlap interact.

2 Sources of Countability

We propose that countability in nouns can be accounted for by combining three factors. Two relate to vagueness, and one to overlap.

The first factor is individuation: what counts as a single quantity (an individual) of a given noun. Individuation can be vague. For some nouns (e.g. *mud*, *blood*) it is vague what a single quantity or individual portion is. There is no generally accepted way to partition the domain of such nouns into individuals. For other nouns (e.g. *cat*, *boy*, *lentil*, *rice*, *kitchenware*, *furniture*) there is little or no vagueness in this respect. There are generally accepted ways to partition these domains (e.g. into single cats, single lentils, single grains of rice). Vagueness of individuation can affect countability. If you can find no one clear way to partition the domain into individuals, there is no clear answer to how many individuals

there are. Individuation features in many accounts of the mass/count distinction ((Grimm 2012), (Rothstein 2010), (Landman 2011) among many others). Vagueness of individuation has been suggested as a factor by Rothstein (2010).

The second factor is quantity: how much of a given amount of stuff is sufficient to classify it as falling under a noun. This can be vague too. For some nouns (*mud*, *blood*, *rice*, *lentil*) a single speck of mud or a single grain of rice is not always sufficient to count as mud or rice (you can truly say "we don't have any rice (for dinner)" when there are only a few grains left in the packet). However, it is vague just how much is enough in a given context. For other nouns (*cat*, *boy*, *kitchenware*, *furniture*), there is little or no vagueness in this respect. Vagueness in quantity can affect countability. If there is no clear minimal quantity of N that counts as N (in a context), then there are no clear minimal parts of N to count.

These two vagueness factors are highly interrelated, but they are separable. Nouns such as *lentil* and *rice* are not vague in the first respect, but are vague in the second. This type of quantity vagueness is suggested as the source of the mass/count distinction in Chierchia (2010).

The third factor is overlap. For some nouns (*furniture*, *kitchenware*), the things classified as individuals/single quantities may overlap. For example, a pestle and mortar can count as a single item of kitchenware, but individually, a pestle and a mortar can each count as single items of kitchenware. For other nouns (*cat*, *boy*, *lentil*, *rice*) there is no overlap in this respect. Overlap can affect countability, because it leads to double counting. Is a pestle and mortar one, two or even three items of kitchenware? The overlap of individual entities (things that count as one) is proposed as the source of the mass/count distinction in Landman (2011).

The first vagueness factor and the overlap factor can interact. If it is vague how to partition a domain into individuals, then it will not be possible to determine what the disjoint non-overlapping individuals are. In some sense, therefore, individuation-vague nouns (*mud*, *blood*) are also overlapping. Hence, nouns can be overlapping as well as vague with respect to both individuation and quantity $[+V, +O]$ (*mud*, *blood*), quantity vague but non-overlapping $[+V, -O]$ (*rice*, *lentil*), overlapping but non-vague $[-V, +O]$ (*kitchenware*, *furniture*), and neither vague nor overlapping $[-V, -O]$ (*cat*, *boy*). As Table 1 helps to show, these four classes display striking cross- and intralinguistic patterns when it comes to the encoding of their members as count $[+C]$ and mass $[-C]$.

	$[+O]$	$[-O]$
$[+V]$	$\text{mud}_{[-C]}$, $\text{blood}_{[-C]}$	$\text{rice}_{[-C]}$, $\text{lentils}_{[+C]}$= $\text{lešta}_{[-C],Bulgarian}$
$[-V]$	$\text{furniture}_{[-C]}$= $\text{huonekalu}_{[+C],Finnish}$	$\text{cat}_{[+C]}$, $\text{boy}_{[+C]}$

Table 1. Feature Matrix for $[\pm V], [\pm O], [\pm C]$

Nouns in the upper left quadrant of Table 1 are typically *substances* and are encoded universally as mass, while nouns in the lower right quadrant are prototypical count nouns, and both types tend not to display cross- or intralinguistic variation in the encoding as count or mass. Nouns in the upper right quadrant

are typically *granular*. Such nouns will display count/mass variation e.g. $rice_{[-C]}$, $lentil_{[+C]}$, $lešta_{[-C],\text{lentil (Bulgarian)}}$. Nouns in the lower left quadrant are typically aggregates in the sense of Payne and Huddleston (2002) and are often superordinate categories. Such nouns display cross- or intralinguistic count/mass variation e.g. $furniture_{[-C]}$, $meubel_{[+C],\text{furniture (Dutch)}}$, $jalkineet_{[+C],\text{PL,footwear (Finnish)}}$.

This leads us to formulating the following Hypothesis: The presence of individuation vagueness entails uncountability ($[-C]$). The absence of vagueness and overlap entails countability ($[+C]$). The presence of quantity vagueness, without overlap allows grammaticization of a given noun concept as $[+C]$ or $[-C]$. The presence of overlap without quantity vagueness allows grammaticization of a given noun concept as $[+C]$ or $[-C]$.

As we shall go on to argue, this variability may ultimately be viewed as a reflection of the differences in perspective on what appears to be the same slice of the real world. Most importantly, no single source account, such as the vagueness-only account of Chierchia (2010) or the overlap-only account of Landman (2011), will be sufficient to account for all the count/mass data as represented in Table 1.

3 Probabilistic Mereological TTR

3.1 Individuation and Individuation Vagueness

The basic schema for a record type involving a nominal predicate will be the record type in (1) where P would be replaced with *cat*, *rice*, *mud* etc. The type ${}^{*}Ind$ is the type of *stuff* (a type involving substances, individuals and any sums thereof):

$$\begin{bmatrix} x & : {}^{*}Ind \\ c_p & : P(x) \end{bmatrix} \tag{1}$$

Following Krifka (1995), we view the application of nominal predicates as being affected by both the *qualitative* and *quantitative* factors. In addition to whatever observable and intrinsic properties that might usually be associated with the qualitative aspects of a noun's denotation, we assume properties such as boundedness, contiguity, size, and topology. For every type as in (1) we assume a dependent type including $P_{qual}(x)$ which packages together all of these properties such as in (2):

$$\begin{bmatrix} x & : {}^{*}Ind \\ c_p & : P_{qual}(x) \end{bmatrix} \tag{2}$$

The quantitative aspect of applying nominal predicates will be characterised as a function $f_{pqual} : (RecType \rightarrow \mathbb{N})$. This is a quantising function which maps Record types, such as (2), to a natural number, where the natural number represents a quantity of P. So, it takes the type of things with rice qualities, boundedness, contiguity, size, and topology, for example, and maps this to a number. (Where 1 would indicate a single rice grain, and, 2

a sum of two rice grains etc.)[1] Finally, we also introduce a condition that the output of the function is some particular value. All of these components together yield a schema for a record type in (3), with an example in (4).

$$\begin{bmatrix} c_{ppt} : \begin{bmatrix} x & : {}^{*}Ind \\ c_{p_{ql}} & : P_{Qual}(x) \end{bmatrix} \\ f_{p_{qt}} : (\begin{bmatrix} x & : {}^{*}Ind \\ c_{p_{ql}} & : P_{Qual}(x) \end{bmatrix} \rightarrow \mathbb{N}) \\ i & : \mathbb{N} \\ c_{p_{qt}} & : f_{p_{qt}}(c_{ppt}) = i \end{bmatrix} \quad (3) \qquad \begin{bmatrix} c_{ppt} & : \begin{bmatrix} x & : {}^{*}Ind \\ c_{rice_{ql}} & : rice_{Qual}(x) \end{bmatrix} \\ f_{rice_{qt}} & : (\begin{bmatrix} x & : {}^{*}Ind \\ c_{p_{ql}} & : rice_{Qual}(x) \end{bmatrix} \rightarrow \mathbb{N}) \\ c_{rice_{qt}} & : f_{rice_{qt}}(c_{ppt}) = 1 \end{bmatrix} \quad (4)$$

The type in (4), with quantity value of 1, represents the type of single grains of rice. For this and other such special cases with quantity values of 1, we introduce a notational convention. Let Ind_{cat}, Ind_{rice} etc. be the type of single cats, (grains) of rice etc. We then abbreviate (4) as (5):

$$[\, x : Ind_{rice} \,] \quad (5)$$

In prob-TTR, judgements are of the form $p(a : T) = k$ where $k \in [0, 1]$. Within Cooper et el's learning model, $p_{A,\mathfrak{J}}(r : T)$ is the probability that an agent, A, assigns to r being of type T with respect to $\mathfrak{J}$ (her experience and learning data set).

Individuation vagueness (which affects substance nouns like *mud*) can be represented as the low probability of anything in an agent's experience being categorised as a single quantity. For example, with *mud*, nothing one has experienced counts as a single mud quantity. This is because there is no principled way to divide the stuff with the right mud qualities into parts. Assuming a *threshold probability* θ, below which an agent will not make a type judgement, this means that for all $r : Rec$ in $A, \mathfrak{J}$:

$$p(r : [\, x : Ind_{mud} \,]) < \theta \quad (6)$$

Nothing the agent has experienced is judged to be a single quantity of mud because for every mud experience, any part of that mud is as good a candidate to be a single unit of mud as the whole is. Intuitively, this is precisely because mud lacks the relevant boundedness, contiguity, size, and topology properties to allow individuation.

3.2 Quantity Vagueness

Even for those nouns that are *not* individuation vague, another form of vagueness (first described by Chierchia 2010) can still remain. In any given situation, if some small amount of P is not enough to count as P, but some larger amount is, it can be vague what the least amount to count as P is. For example, one lentil or rice grain left on a plate does not mean that you have not eaten (all)

[1] The quantising function may be more course grained for higher values, just as perception is. For example, for *rice* high quantity values of could represent some range of numbers of grains.

your rice/lentils. Other nouns are not, or at least, are not usually vague in this way. A single chair is usually enough to count as furniture. A single cat is usually enough to count as a cat. For individuation vague nouns (which have no clear single units), quantity vagueness can also be present: A tiny fleck of mud left on your boot does not mean that you have not cleaned (all) the mud off your boots, but it can be vague how much mud would.

This form of vagueness can be represented as gradience in a conditional probability distribution. For example, $p(x \text{ is rice} \mid x \text{ is some quantity } i \text{ of rice})$ will be low for small quantities, higher for higher quantities. We get a sorites-like slope, where, even if there is a threshold probability for assertion, an agent may not be sure whether or not she is at the threshold. In probM-TTR, this will be represented as the probability distribution generated by a record being of a type as in (1), given that it is of a type as in (3) (with i as a variable). For *rice*, this would be represented in (7)

$$p_{A,\mathfrak{J}}(r : \begin{bmatrix} x & : {}^*Ind \\ c_{rice} & : rice(x) \end{bmatrix} \mid r : \begin{bmatrix} c_{ppt} & : \begin{bmatrix} x & : {}^*Ind \\ c_{rice_{ql}} & : rice_{Qual}(x) \end{bmatrix} \\ f_{rice_{quant}} & :(\begin{bmatrix} x & : {}^*Ind \\ c_{p_{ql}} & : rice_{Qual}(x) \end{bmatrix} \to \mathbb{N}) \\ i & : \mathbb{N} \\ c_{rice_{qt}} & : f_{rice_{quant}}(c_{ppt}) = i \end{bmatrix}) = k \quad (7)$$

So, the probability k that something is *rice*, given that it is i quantised portions of rice (roughly, i number of grains, although see footnote 1), varies with the value of i. There will be little or no such vagueness for nouns such as *cat*, *furniture*, etc. This is represented as an above the threshold probability of something being a cat, given it is a single cat (8), and an above the threshold probability of something being furniture, given that it is a single (item of) furniture (9).

$$p_{A,\mathfrak{J}}(r : \begin{bmatrix} x & : {}^*Ind \\ c_{cat} & : cat(x) \end{bmatrix} \mid r : \left[x : Ind_{cat} \right]) > \theta \quad (8)$$

$$p_{A,\mathfrak{J}}(r : \begin{bmatrix} x & : {}^*Ind \\ c_{furn.} & : furniture(x) \end{bmatrix} \mid r : \left[x : Ind_{furn.} \right]) > \theta \quad (9)$$

The difference between nouns like *cat*, and nouns like *furniture* will be that single cats (entities of type $[x : Ind_{cat}]$) do not overlap, but single items of furniture (entities of type $[x : Ind_{furn}]$) may well do.

3.3 Overlap

We will capture overlap and non-overlap in terms of *Disjointedness*: There is overlap within a type if the type is not disjoint. A type T is disjoint iff for all $a : T$ and $b : T$, if $a \neq b$, then $a \cap b = \varnothing$. The type of single items of furniture ($[x : Ind_{furn}]$) is not disjoint, since all of the following are of this type: *dressing-table*, *mirror*, *dressing-table* $\cup$ *mirror*.

Interpreted probabilistically, we will again use the probability threshold for type judgements, θ:

Definition 1. T is Probabilistically Disjoint relative to θ

$T : Disj_\theta$ *iff there is at least some* a *such that* $p(a : T) \geq \theta$*, and for all* a, b *such that* $p(a : T) \geq \theta$ *and* $p(b : T) \geq \theta$*, if* $a \neq b$ *then* $a \cap b = \varnothing$

So T is disjoint relative to threshold θ iff T is not empty from the perspective of sufficient certainty of what is of type T, and no two things which are sufficiently probably of type T overlap.

4 Countability in probM-TTR

Within the formal representations of nouns we have developed here, the denotations of nouns admit of two different *perspectives*: The "zoomed out" perspective of what counts, with sufficient certainty, as the smallest amount of stuff which is of some type (as determined by conditional probability distributions like in (7) for *rice*); and the "zoomed in" perspective of what counts as single, *but also disjoint* entities of a noun (disjoint entities of the type Ind_p). From each perspective, countability may be prevented by either vagueness, which prevents the identification of what one should count, or overlap, which leads to double counting and so incompatible counting results. We will now derive our four classes of nouns via whether countability is prevented from one of these two perspectives, from both, or from neither.

Prototypical count nouns: For nouns such as *cat*, the zoomed in and the zoomed out perspectives coincide. Zooming out, the smallest quantities that clearly satisfy the predicate *cat* are single individual cats, as the probability of something being a cat, given that it is of type Ind_{cat}, is very high/above the threshold. Zooming in, we see that this type Ind_{cat} is also disjoint. Either way, we are left with just individual cats, and so only one possible counting result. The fact that we get the same result from either perspective can explain why non-vague, non-overlapping nouns are encoded universally as count.

Vague, non-overlapping nouns: For nouns such as *rice* and *lentils*, the zoomed in and the zoomed out perspectives do not coincide. Zooming out, it is (quantity) vague what the smallest entities are that satisfy the predicates *rice* and *lentils*. Single grains/lentils frequently do not count as satisfying these predicates, as the probability of something being rice/lentils, given that it is of type Ind_{rice}/Ind_{lentil} is very low/below the threshold. Large quantities do count frequently enough, but the cut-off point is vague. From this zoomed out perspective we cannot focus on a reasonable counting base and so countability is blocked. Zooming in, the types Ind_{rice} and Ind_{lentil} are anyway disjoint (single grains/single lentils do not overlap). Furthermore, Ind_{rice} and Ind_{lentil} are not (particularly) vague. So counting can occur at this level.

The difference in the result between perspectives can explain why we see mass/count variation in such nouns. Depending on whether their denotations are

viewed from a zoomed in or a zoomed out perspective, we can either count or fail to be able to count what is there. On this understanding of the mass/count distinction, English, for example, conceptualises *rice* from a zoomed out perspective, and *lentils* from a zoomed in perspective, hence the difference in count/mass encoding. Bulgarian conceptualises *lešta* (lentil) from a zoomed out perspective, hence *lešta* is encoded as mass.

Non-vague, overlapping nouns: For nouns such as *furniture* and *kitchenware*, the zoomed-in and the zoomed-out perspectives also do not coincide, but not for the same reasons as with nouns such as *rice* and *lentils*. Zooming out, there is no quantity vagueness for furniture or kitchenware. The smallest quantities that clearly satisfy the predicates $furniture$, $kitchenware$ are items of kitchenware and items of furniture respectively (the probability of something being a kitchenware/furniture, given that it is of type $Ind_{furn.}$/$Ind_{kitch.}$ is very high/above the threshold). However, we cannot count entities of the types $Ind_{furn.}$ and $Ind_{kitch.}$, because they are not disjoint. This overlap leads to incompatible counting results. For example $\{pestle, mortar, pestle \cap mortar\}$ are all of type $Ind_{kitch.}$, but it is not, from this perspective decided whether this set constitutes one, two or even three items of kitchenware. Zooming in, we are forced, relative to a context, to try to find a disjoint subset of the set $Ind_{kitch.}$ and thereby ignore/remove the overlap. The type Ind_{kitch} is not vague and so it is clear what is of this type. Taking the above example, this allows two possible contexts. In context 1, we take the disjoint subset that leaves $\{pestle, mortar\}$ in which case there are two things. In context 2, we take the disjoint subset that leaves $\{pestle \cup mortar\}$ in which case there is only one thing.

The different outcome of the two perspectives can explain why we see mass/count variation in such nouns. In English, for example, both *kitchenware* and *furniture* are viewed from a zoomed out perspective and so are encoded as mass. In Finnish $huonekalu_{[+C],\text{furniture}}$ is viewed from a zoomed in perspective and is encoded as count.

Vague and overlapping nouns: For nouns such as *mud* and *blood*, the zoomed in and the zoomed out perspectives *do* coincide. Zooming out, it is (quantity) vague what the smallest entities are that satisfy the predicates *mud* and *blood*. Tiny amounts of mud and blood most of the time do not count as satisfying these predicates, but in contrast to nouns such as *rice* and *lentils*, there is also individuation vagueness here. Within the denotations of *mud* and *blood*, we are not even sure what is of the types Ind_{mud} or Ind_{blood}. From this zoomed out perspective we cannot focus on a reasonable counting base and so countability is blocked. Zooming in does not get us much further. Because of individuation vagueness, we are not sufficiently sure that anything is of type Ind_{mud} or Ind_{blood} etc. That means that types such as Ind_{mud} and Ind_{blood} are not probabilistically disjoint because nothing has a high enough probability of being of these types. That means that we can not discern a disjoint subset within this set. Countability is prevented from a zoomed-in perspective too. From either perspective, there is

no counting result for individuation vague nouns. This can explain why substance nouns always get encoded as mass.

5 Conclusion

Depending upon the ways we can conceptualise some referent of a noun, counting either gets a free pass, faces a conceptually avoidable stumbling block (vagueness or overlap), or is stopped in its tracks (vagueness and overlap). We have proposed that in the first case, nouns will be universally encoded as count. In the second, depending on how the stumbling block was treated within and between languages, we expect mass/count encoding to vary. In the third case, we expect universal mass encoding. If correct, we improve upon Landman (2011) and Chierchia (2010) by accounting for cross- and intralinguistic variation, while also motivating the stubborn resistance that prototypical mass nouns show to counting.

References

Chierchia, G.: Mass nouns, vagueness and semantic variation. Synthese 174, 99–149 (2010)

Cooper, R.: Type theory and semantics in flux. In: Kempson, R., Fernando, T., Asher, N. (Eds.), Philosophy of Linguistics, Handbook of the Philosophy of Science. Elsevier, 271–323 (2012)

Cooper, R., Dobnik, S., Lappin, S., Larsson, S.: A probabilistic rich type theory for semantic interpretation. Proceedings of the EACL 2014 Workshop on Type Theory and Natural Language Semantics (2014)

Grimm, S.: Number and Individuation. PhD Dissertation, Stanford University (2012)

Krifka, M.: Common nouns: A contrastive analysis of English and Chinese. In: Carlson, G., Pelletier, F. J. (Eds.), The Generic Book. Chicago UP, 398–411 (1995)

Landman, F.: Count nouns – mass nouns – neat nouns – mess nouns. In: The Baltic International Yearbook of Cognition: Vol. 6. pp. 1–67 (2011)

Payne, J., and Huddleston, R.: Nouns and Noun Phrases: In: Pullum, G. and Huddleston, R. (Eds.) The Cambridge grammar of the English language. Cambridge UP, 323–524 (2002)

Rothstein, S.: Counting and the mass/count distinction. Journal of Semantics 27 (3), 343–397 (2010)

Are Widows Always Wicked? Learning concepts through enthymematic reasoning

Ellen Breitholtz

University of Gothenburg, Department of Philosophy, Linguistics and Theory of Science
ellen@ling.gu.se

Abstract. This paper suggests that enthymematic reasoning may play a role in the acquisition of new concepts by a language learner. To illustrate this we analyse an example of natural dialogue using Type Theory with Records. Our approach allows us to represent misunderstanding and misinterpretation of meaning, since it is based on the conceptualisation of entities in individuals rather than on a God's eye view of meaning. Thus our account fits well with an approach to word meaning where speakers are constantly adjusting meanings on the basis of experience. An enthymematic perspective on the learning of concepts relates previous work done on enthymematic reasoning cast in TTR to work where enthymemes are used to explain lexical disambiguation.

Keywords: Enthymeme, topos, Type Theory with Records, learning, lexicon, human reasoning

1 Introduction

Consider the example in (1) of an interaction between a mother and a child:

(1)
Mother: When Snow White was still a baby her mother died. After some time her father, the king, remarried. His new wife was beautiful but vain and wicked.
Child: Yes mum – a widow!

To anyone familiar with the conventional meaning of the word *widow*, it seems obvious that the child has got it wrong. However, how did she get it wrong? And can the mechanisms of how she got it wrong explain how most of us eventually get it right?

In this paper we suggest how reasoning using *enthymemes* – rhetorical arguments – can be a means not only of lexical disambiguation (as suggested by Pustejovsky (1998)), but also a means of acquiring new concepts. We will introduce enthymematic reasoning in dialogue as it is presented in Breitholtz (2014a),

Breitholtz (2014b), Breitholtz and Cooper (2011). We will then analyse the example in (1) in terms of the inferences that may be drawn using enthymematic reasoning based on *topoi* in the resources of the dialogue participants. For our account we will use Type Theory with Records (TTR) Cooper (2013), Cooper (2005), Ginzburg (2012).

2 Enthymematic Reasoning

2.1 What Is an Enthymeme?

An *enthymeme* is a rhetorical argument which appeals to what is in the listener's mind. In his *Rhetoric* Aristotle refers to enthymemes as belonging to the *logos*-part of discourse, that is, to the part concerned with content and reasoning. Aristotle also relates the enthymeme to logic by calling it a "rhetorical syllogism". However, in the case of a syllogism, the inference presented as the conclusion is non-negotiable, while an enthymeme owes a lot to context and background knowledge. It is therefore often negotiable and defeasible.

Jackson and Jacobs (1980), Breitholtz (2014a), Breitholtz and Cooper (2011) show that enthymematic reasoning occurs frequently in spontaneous dialogue and other types of highly interactive language genres, not only in rhetoric In 2, *A* presents an argument that she cannot make it to a party because she is going to a wedding, but since the bride is pregnant she might be able to drop by later in the evening.

(2) *A*: Oh! I'm invited to a wedding that night. But the bride is pregnant so I might drop by in the wee hours. (**?**)p1]Breitholtz2014b

Thus, *A* implies that the bride being pregnant might be a reason for the wedding not continuing very late – an assumption itself underpinned by a chain of inference. We refer to all of these underpinnings as the *topoi* of the enthymeme.

Ducrot (1988) and Anscombre (1995) propose a theory of meaning in context where the notion of topos is central. The theory is based on the idea that between two utterances A and C, where one of them is an assertion or a suggestion, exhortation, etc. and the other an assertion which functions as a support for the first, there is always a link which sanctions the interpretation of A and C as an (enthymematic) argument. Important points of the theory they propose are that topoi are assumed or taken for granted in a community, and that one topos can be employed in various contexts as underpinning for different types of arguments, but also the converse – that different topoi can be employed in one context.

Anscombre suggests that topoi are part of *ideology*, ways in which we perceive the world, and ideologies are not monolithic. Therefore, a principle like *opposites attract* and *birds of a feather flock together* may co-exist not only in one community, but in the set of topoi of one individual, and even be applicable in the same contexts. Following Ducrot and Anscombre, we argue that topoi,

contrary to logical rules, do not constitute a monolithic system. Instead the system of topoi in the resources of one individual consists of principles which may be combined in different ways – like logical rules – but which may be inconsistent if combined in a specific situation.

2.2 Enthymemes and the Lexicon

The work on enthymemes and topoi by Breitholtz, Ducrot and Anscombre is mainly concerned with the relation between utterances in dialogue and discourse. However, Pustejovsky (1998) suggests that enthymemes are also important for lexical interpretation. He discusses the example *Steven King began a new novel* (**?**)p 238]Pustejovsky1998 and argues that the world knowledge regarding who Steven King is (a writer) and what writers do (write novels) serves as hidden premises in an enthymematic argument with the conclusion that Steven King has begun *writing* a new novel. In the terms of Ducrot and Breitholtz, this example is underpinned by a *topos* saying that "writers write", or "if someone is a writer, they write". However, there are also other possible topoi in our resources that could be drawn on in this context, for example the topos "people read novels". Interestingly, our bias towards interpreting *began* in this case as *began writing* rather than *began reading* is not directly linked to the number of occurrences of events where writers write novels compared to the number of occurrences of events where writers read novels – arguably, writers on average read more novels than they write. Instead the preference for the first interpretation has to do with the salience of the topos.

3 Learning concepts using enthymemes

We will now return to the example dialogue in (1). We will consider the types representing the individuals mentioned in the data, the enthymematic arguments present and the topoi that underpin the arguments or are obtained as a result of generalising an enthymeme. We would like to think of reasoning in terms of dialogue game boards representing dialogue participants' respective take on the dialogue at any point in time. We should therefore think of the enthymematic arguments, topoi and types representing individuals as features of such a gameboard. Following many other gameboard-based approaches to dialogue, for example the ones found in Ginzburg (2012) and Cooper (2013) we use the formalism Type Theory with Records. TTR is a rich type theory that has the advantage of modelling both utterance events and utterance types, which is crucial for analysing meta-communicative aspects of interaction. Subtyping in TTR is also important for our account of how we employ topoi in different enthymemes through operations like restriction and generalisation. The mother's utterance in (1) says explicitly that Snow White's stepmother is vain and wicked. Thus we may assume that the type of Snow White's step mother that is common ground in the dialogue this far is the one in (3)[1].

[1] This could also be a subtype of 3 including other constraints like "woman". However, we leave aside such considerations here for clarity

(3) $$\begin{bmatrix} \text{x=Snow White's stepmother:}\mathit{Ind} \\ \text{c}_{\text{vain}}\text{:vain(x)} \\ \text{c}_{\text{wicked}}\text{:wicked(x)} \end{bmatrix}$$

The child's utterance together with that of the mother, makes up a co-constructed enthymeme saying that *Snow White's stepmother is vain and wicked, therefore, she is a widow*. We represent topoi and enthymemes as dependent types, functions from records to record types, as seen in (4). Intuitively, such a function represents the idea that if you have a situation of one type – say, the type where someone is vain and wicked – then you can predict the type of situation where that person is a widow.

(4) $$\lambda r\text{:}\begin{bmatrix} \text{x=Snow White's stepmother:}\mathit{Ind} \\ \text{c}_{\text{vain}}\text{:vain(x)} \\ \text{c}_{\text{wicked}}\text{:wicked(x)} \end{bmatrix}.\ \left[\text{c}_{\text{widow}}\text{:widow}(r.\text{x})\right]$$

For the child to make this argument, we must assume that she has access to a topos that underpins the enthymeme. To obtain a possible topos we generalise the enthymeme in (4) by removing the value of the manifest field (x=Snow White's stepmother:*Ind*)in the domain type. We now have a dependent type representing a topos saying that "if someone is vain and wicked, they are a widow".

(5) $$\lambda r\text{:}\begin{bmatrix} \text{x:}\mathit{Ind} \\ \text{c}_{\text{vain}}\text{:vain(x)} \\ \text{c}_{\text{wicked}}\text{:wicked(x)} \end{bmatrix}.\ \left[\text{c}_{\text{widow}}\text{:widow}(r.\text{x})\right]$$

Now, we may ask, how was this topos established? It seems reasonable to say that if the child believes that someone being vain and wicked is a reason for concluding that that person is a widow, the child perceives *vain and wicked* as essential components of the meaning of the word *widow*. But how was this idea established? No one is likely to have told the child that *widow* means *vain and wicked*, or that widows are vain and wicked. We argue that the child has learnt this from reasoning.

Consider for example this excerpt from another fairy tale – *Cinderella*:

(6) "After a few years Cinderella's father took a new wife, a widow with two daughters of her own"

From this passage, we learn that Cinderella's stepmother is a widow. As the story evolves, we get multiple proof that she is also vain and wicked. From the story of *Cinderella* a topos regarding widows may be tentatively construed, namely the one in (7):

(7) $$\lambda r\text{:}\begin{bmatrix} \text{x:}\mathit{Ind} \\ \text{c}_{\text{widow}}\text{:widow(x)} \end{bmatrix}.\ \begin{bmatrix} \text{c}_{\text{vain}}\text{:vain}(r.\text{x}) \\ \text{c}_{\text{wicked}}\text{:wicked}(r.\text{x}) \end{bmatrix}$$

This topos says that if we have a situation of the type where someone is a widow, we also have a type of situation where that person is vain and wicked.

Now, from dependent types like topoi and enthymemes, we may derive *fixed point types*. A fixed point type in this context would be a type describing the situation from a more holistic perspective, so instead of perceiving that we have access to a topos saying that if someone is a widow, they are vain and wicked, we perceive a scenario where this topos would be accuarate as one type of situation. For example, if we have a topos saying that in the summer, people spend time at the beach, we may from this topos construe a type of situation In which the topos would be realised, namely a type of situation where it is summer and people spend time at the beach. In formal terms this means that if τ_2 is the topos in (7), then a fixed-point type for τ_2 is a type T such that $a : T$ implies $a : \tau_2(a)$. We can obtain such a type by merging the domain type and the result type adjusting the references to r in the dependencies, as in (8). For a thorough discussion of fixed point types in TTR, see Cooper (fthc).

(8)
$$\begin{bmatrix} \text{x}{:}\mathit{Ind} \\ \text{c}_{\text{widow}}{:}\text{widow(x)} \\ \text{c}_{\text{wicked}}{:}\text{wicked(x)} \\ \text{c}_{\text{vain}}{:}\text{vain(x)} \end{bmatrix}$$

We perceive the reasoning with enthymemes and topoi to be partly associative, and related to the kind of associations that arise in neural activity where association of two patterns of neural activation during perception eventually lead to the fact that an external stimulation of one pattern will engender the second pattern even in the absence of a stimulus. This means that for a child to perceive a number of qualities co-ocurring in one individual in a situation, the child would find it reasonable to construe various types of dependencies between these qualities. Thus, when encountering a type of situation where someone (Snow White's stepmother) is vain and wicked, the child draws on a topos construed from the type of situation in (8), saying that if someone is vain and wicked, they are a widow.

Other topoi that would be possible to derive from (8) would be ones saying that vain widows are wicked and that widows are wicked and vain. In fact, from all topoi we may derive any topos that has the same fixed point type. The more constraints we add to the situation type encountered by the child in *Cinderella*, such as "in_fairytale" and "stepmother", the more acceptable some of the dependent types construed from the situation type will seem.

However, it is not the case that the topoi we have access to are symmetrical in the sense that the dependency is equally strong in both directions. For example, it seems like a useful rule of thumb that someone who is called Lisa identifies as a woman, while the inference rule that if someone identifies as a woman, then that person is called Lisa, would be a pretty useless rule. In order to acchieve a dynamic theory of language learning and reasoning, we would thus need to include a probabilistic component. Exactly how this would be set up we

leave aside for the time being, but it seems intuitively clear that the child, as it encounters the word *widow* in other situations, will revise the dependencies of the relevant topoi in her resources, and *vain* and *wicked* would gradually move from the centre of the meaning of *widow* to the periphery to become at most a connotation.

4 Conclusion

We have seen an example that illustrates how we use reasoning to establish the meaning of words as well as to disambiguate word meaning. Our approach allows us to represent misunderstanding and misinterpretation of meaning, since it is based on the conceptualisation of entities in individuals rather than on a God's eye view of meaning. Thus our account fits well with an approach to word meaning where speakers are constantly adjusting meanings on the basis of experience, which can be found in work by for example Cooper (2012), Kempson *et al.* (2012), Ludlow (2014) and Pustejovsky (1998).

One of the advantages of using topoi as the underpinning for the kind of non-monotonic reasoning we find in enthymemes, rather than default rules, is that the set of topoi of one agent does not constitute a monolithic logical system. Thus they do not need to be consistent or lead to consistent conclusions even within one model or domain (Breitholtz, 2014a). This ability to follow various strains of reasoning also inconsistent ones seems to be a prerequisite for the complex type of interactive language understanding and problem solving that humans master so well. However, in order to fully take advantage of the possibility to model this ability, we need to be able to account for the reasoning of agents with access to a wider range of topoi than those we have considered here. A natural progression of the account presented here would be to extend the model presented in Breitholtz (2014a) to include a probablilistic component. This would enable us to make predictions regarding the enthymematic inferences of an agent with access to several topoi applicable in a particular situation. It would also allow for modelling the learning of new topoi through interaction with other agents. Interesting work has been done by Cooper *et al.* (2014) on probabilistic semantics in TTR, and Clark and Lappin (2010) convincingly show how language learning is related to probability theory. Both of these approaches fit well with the approach that we suggest, and they thus offer a way to introduce a probabilistic component into the account of learning presented here.

Bibliography

Anscombre, J.-C. (1995). La théorie des topoi: Sémantique ou rhétorique? *Hermés*, **15**.

Aristotle, G. A. K. (2007). *On Rhetoric, a theory of civic discourse*. Oxford University Press.

Breitholtz, E. (2014a). *Enthymemes in Dialogue: A micro-rhetorical approach*. Ph.D. thesis, University of Gothenburg.

Breitholtz, E. (2014b). Reasoning with topoi - towards a rhetorical approach to non-monotonicity. volume Proceedings of the 50:th anniversery convention of the AISB, pages 190–198. AISB.

Breitholtz, E. and Cooper, R. (2011). Enthymemes as rhetorical resources. In D. D. K. G. E. K. A. S. Ron Artstein, Mark Core, editor, *Proceedings of*, volume SemDial 2011 (LosAngelogue) Proceedings of the 15th Workshop on the Semantics and Pragmatics of Dialogue, pages 149–157.

Clark, A. and Lappin, S. (2010). *Linguistic Nativism and the Poverty of the Stimulus*. John Wiley & Sons.

Cooper, R. (2005). Records and record types in semantic theory. *Journal of Logic and Computation*, **15**(2), 99–112.

Cooper, R. (2012). Type theory and semantics in flux. In R. Kempson, N. Asher, and T. Fernando, editors, *Handbook of the Philosophy of Science*, volume 14: Philosophy of Linguistics. Elsevier BV. General editors: Dov M Gabbay, Paul Thagard and John Woods.

Cooper, R. (2013). Clarification and generalized quantifiers. *Dialogue and Discourse*, **4**(1).

Cooper, R. (fthc). *Type theory and language - From perception to linguistic communication*. Draft, June 17 2014.

Cooper, R., Dobnik, S., Lappin, S., and Larsson, S. (2014). A probabilistic rich type theory for semantic interpretation. In *Proceedings of the EACL 2014 Type Theory and Natural Semantics Workshop (TTNLS)*. EACL.

Ducrot, O. (1988). Topoï et formes topique. *Bulletin d'études de la linguistique française*, **22**, 1–14.

Ginzburg, J. (2012). *The Interactive Stance: Meaning for Conversation*. Oxford University Press.

Jackson, S. and Jacobs, S. (1980). Structure of conversational argument: Pragmatic bases for the enthymeme. *Quarterly Journal of Speech*, **66**(3), 251–265.

Kempson, R., Gregoromichelaki, E., and Cann, R. (2012). Context and compositionality: the challenge of conversational dialogue. *Philosophical and Formal Approaches to Linguistic Analysis*, page 215.

Ludlow, P. (2014). *Living Words: Meaning Under determination and the Dynamic Lexicon*. Oxford University Press.

Pustejovsky, J. (1998). *The Generative Lexicon*. Bradford Books. MIT Press.

The Relative Complexity of Constraints in Co-Predicative Utterances

Bruno Mery[1]

IUT de Bordeaux, Université de Bordeaux, France & LaBRI — CNRS
bruno.mery@me.com

Abstract– We explore the particular variations of semantic felicity in co-predicative utterances the constraints on the combination of facets of polysemous words, and the possible adaptations that can be proposed to existing formal frameworks that support lexical semantics. As the linguistic data is incomplete and disputed, we also include a proposal for a linguistic survey aimed at clearing up many outstanding issues.[1]

1 The Generative Approach to Lexical Semantics and Polysemy

1.1 The Issue of Polysemy in Compositional Semantics

The inarguable facts that words in human languages can be employed with many different meanings, according to the context they are used in, is seemingly at odds with the principle of compositional semantics inherited from Montague, which presupposes that singular meaningful term can be provided for every lexical item. This can be resolved by Word Sense Disambiguation techniques (associating the most probable meaning to each word before composition, using data refined from corpora). Another approach is to hold polysemy to be an important feature of language, and to re-design the compositional semantics around it. In this approach, the lexicon becomes much more complex than a map of words to meanings, as each lexical entry can be combined with others in order to select the appropriate sense for a word used in a specific context: this is the so-called Generative approach to lexical semantics developped in [Pus95] and many subsequent works. We will discuss a few basic features of this approach in Section 1.2.

In the present paper, we will study the specific issue of the felicity of co-predicative utterances. Those constructs are detailed in Section 2.1, the issues of felicity in Section 2.3, and we will focus on the characterisation of constraints on felicity in Section 3, and the necessary adaptations of formal semantics in Section 4.

[1] The present abstract is intended as a discussion material, detailing a specific issue in order to foster collaborations around the subject. Having been prepared by the author in his spare time, it is, by necessity, incomplete. The work presented here is closely related to ΛTY_n, a formal framework for lexical semantics developed since 2006 with Christian Bassac, Richard Moot, Christian Retoré and many other researchers at LaBRI and LIRMM. As a team, we would like to thank everyone that has joined in TYTLES and engaged in discussions and research around type-theoretical lexical semantics.

1.2 GL Qualia, Dots, Processes and Uses

[Pus95] and subsequent works such as [Ash11] have detailed the implications on Montagovian semantics of *relational* polysemy, distinguishing (among many other phenomena) four kinds of meanings that are distinct, but directly related (by opposition to *accidental* polysemy, which consists of different words that happen to be homonyms).

The *qualia* are derived from the "modes of explanation" of Aristotle; the idea is that a noun for something can refer to :

1. The thing itself (formal quale, as in *long sword*),
2. A salient part of the thing (constitutive quale, as in *dull sword* $\rightarrow$ *dull edge*), and, for artificial items,
3. Their creation process (agentive quale, as in *master's sword* $\rightarrow$ *sword made by a master blacksmith*),
4. Their use, for objects that have one (telic quale, as in *fine sword* $\rightarrow$ *sword fine for fighting*).

Pustejovsky argues that such uses are extremely common, and that a competent speaker of a language has access to a qualia, thus justifying that such information should be part of the lexicon itself.

Complex objects, called *dot-objects* (sometimes written •-objects) are words that denote a singular concept that can be envisioned on two or more different *facets*. Canonical examples include books (with physical and informational facets), meals (with an event and a food facet), newspapers (with an organisational and a physical aspect), etc. There are many examples which seem to stem from compounding essential and existential information. As the number and type of facets is not determined, contrary to qualia, and as those facets enjoy differentiated individuation conditions, they have proved to be one of the most serious difficulties for formal models of lexical semantics.

In addition, some implicit processes can shift the meaning of a word referring to something to different states: *grinding* can turn materials into artefacts, animals or plants into food (*delicious salmon*), *packaging* can provide implicit containers for masses (*there is water and wine on the table*) – see [MMR15] for details. Deverbal nouns are notoriously polysemous between process and result (*the construction is at the end of the street*) – see [Jac01], [RCR13].

Finally, additional facets can be bestowed upon a lexical item via explicit constructions, which are easier to account for. They include *as*-phrases (*I do not have issues with Mr. X as a lawyer; however, as a candidate...*) and *for*-phrases (*this knife is sharp enough for shaving*).

2 The Many Facets of Co-Predicative Sentences

2.1 Co-Predications

Co-predicative utterances, phrases or sentences, or *co-predications* for short, consist in the explicit reference to two (or more) different facets of the same lexical items at the same time. The most classical example is *heavy and interesting book*, but there are many others. The difficulty of co-predications is that the two predicates apply to different types (here, to physical and informational objects respectively). Lexical semantics frameworks that do not consider this and simply coerce the type of the argument to the one required by the predicate fail to provide a suitable typing for *book* in such sentences.

2.2 Mixing Polysemy Facets

In the original formulation by Pustejovsky, co-predications were mostly studied between facets of dot-objects. However, they can be equally valid between various sources:

- *two or more qualia*:
  - *good, expensive wine* (telic+formal);
  - *fast blue car* (constitutive+formal)...;
- *two or more facets* of a dot-object:
  - *red closed door* (physical+aperture);
  - *liberal, picture-less tabloid* (organisation+physical)...;
- *a facet of a dot-object and a qualia of another facet*:
  - *I have an inspired article in my briefcase* (agentive quale of the informational facet + physical facet);
- *process, result*, and other facets:
  - *the translation, printed right after its lengthy completion, is considered bold yet naive* (physical and agentive quale of the informative facet of the result + process);
- *an arbitrary number of references* to different facets in a coherent discourse:
  - *The book is 5lbs, and has 400 pages. It has been set in Times 12, with large margins. Its leather cover is aged but sound. Its writing took five years, and the completion of the hundred in-quarto printings four months. [...] It certainly is an interesting read.*

2.3 Felicitous and Infelicitous Co-Predications

There are thus many possible combinations of facets that can be predicated on simultaneously. However, some co-predications are infelicitous. We have elaborated previously on the case of grinding :

* *The salmon was lightning-fast and delicious*,

many examples of process/result alternations:

* *The construction took three months and stands tall*,

as well as capital-governments alternations:

* *Washington is an old American city and has denounced Teheran*.[2]

[2] Those examples are subject to personal interpretation and linguistic idiosyncrasies that we will discuss in Section 3.3.

3 Characterising Constraints on Felicity

3.1 Relaxable and Fixed Constraints

The most evident constraints are the following: facets that can cognitively co-exist for the same object are not constrained in their co-predications, while facets that are only present after a transformative process (grinding, packaging, resultatives) are exclusive to all others.

In addition, a proper name of a polysemous entity (such as a city or place) can be used as a paraphrase for a specific group of people (such as the name of the capital being a proxy for the national government), to the exclusion of all other possible senses. We can thus distinguish between two rough classes of facets: *flexible* and *rigid*.

However, the rigidity constraint can be relaxed in some cases. A syntactic split of the components of the co-predications can render some infelicitous phrases acceptable : *the salmon, which was very fast, is delicious*, or *the construction, that took three months to complete, stands tall*. Beyond the syntactical separation, the use of different tenses or explicit contrast using *yet* or *but* can also relax the constraints on the use of facets that correspond to different, exclusive states of the same entity.

Capital/government (and related) alternation constraints are much more difficult to relax, considering: *Washington is an old American city. It is located on the banks of the Potomac. * It has denounced Teheran.*.

That distinction has given us the three classes of constraints that we have been using until now, *flexible* facets, *semi-rigid* facets (the constraints can be relaxed via syntactic means) and *rigid* facets (the constraints are fixed).

3.2 City Names: a Short Case Study

However, this does not cover the whole complexity of constraints on the felicity of co-predications. See [RCR13] for a case study on deverbal co-predications; we will discuss here the specific case of city names.

City names are highly polysemous, with many possible facets that can be separated in two groups. The first is composed of facets that pertain to the city as a whole: architecture, location, size, atmosphere, climate, population, lifestyle... that we will call *characteristics*, and the second, of facets we will call *essentials*, that use the name of the city to refer to a specific group of people, such as city/metropolis/regional/national government and sport clubs. In our first approximation, characteristics are flexible and essentials are rigid.

However, the following constructions can allow some of the essentials to co-predicate:

- *lexical proximity* – if two facets share the same lexical field, as in *Barcelona dominates Europe in football as well as in handball*;
- *object identity* – if the predication is made on a single direct object, as in *Barcelona dominates Europe in football as in architecture*;
- *zeugma* – if zeugma are considered acceptable, as in *Barcelona dominates Europe in football as they do Madrid in politics*;

– *discourse flow* – logical relations in the narration can provide felicitous readings, as in *Detroit, by lake Michigan, has filed for bankruptcy; the city has become desolate and lifeless*.
– The above can be daisy-chained in order to co-predicate over seemingly incompatible facets in a long discourse: *Bordeaux, struggling with traffic issues due to its growing population and its geographical position on both sides of the Garonne, wishes to build a new bridge in order to ease commuting between those*.

3.3 Evaluating the Idiosyncrasies of Felicity in Co-Predications

We want to stress that felicity is an highly subjective notion, that varies from speaker to speaker. Additionally, it appears that some of the constraints are idiosyncratic; anecdotical evidence, including separate personal communications from Asher, Lecomte and Luo indicate that some common co-predications are felicitous in English but not in Chinese. While this should not stop formal frameworks to develop mechanisms that integrate constraints on co-predications, establishing a robust catalogue of the actual usages is important.

We would like to propose a set of templates for sentences that can be adapted to valid local city data, with co-predications we consider felicitous, infelicitous, and forced by the various mechanisms evoked above, as well as straightforward predications on a single facet as controls. This survey should be conducted by native speakers of several languages in different linguistic groups (minimally Romance, Germanic and East-Asiatic), and we would like it to be part of collaborative international projects that are being constructed as a result of the recent interest in lexical semantics.

4 A Proposal for Linear Types and Terms

Constraints on co-predications can be added to current lexical frameworks as an external mechanism[3]. However, it is also possible to allow for every possible combination with a redesign of the formalism. Our proposal incorporates a λ-calculus whose terms are typed with formulae of the second-order linear intuitionistic logic.

Linear ΛTY_n is adapted from our framework in the following way:

– Simple predications are implemented by linear application (*heavy* is $H^{\varphi \multimap \mathbf{t}}$).
– Basic morphisms (type transformations) also (selecting the physical facet is done via $\Lambda\alpha.f^{\alpha \multimap \varphi}$).
– Lexical transformations (giving access to specific facets of each term) are implemented as pairs, the first component being an accessor to the facet (a type transformation, as above) and the second assessing the compatibility of the facet with other transformations.

[3] It is sufficient to keep track of the flexible/rigid characteristic of terms, relaxing the constraints when appropriate using a set of rules that detect syntactic or discursive features, and to stop the composition when incompatible co-predications are detected; this is the approach we have proposed so far, see [Ret14] for a recent synthesis in our ΛTY_n framework.

– Additional terms, that modify this compatibility, can be provided by syntax, discourse and pragmatics in order to relax the constraints when appropriate.

A detailed account of this proposal is given in [Mer15].

Summary

While the work presented here is still in progress, we are convinced that the issue of constraints on co-predication is of importance in the establishment of precise natural language semantics. We hope to gather enough linguistic data in order to characterise the extent of the phenomena, and have the necessary formal tools to treat it accurately. Our goal remains to integrate issues of lexical semantics and polysemy directly in the flow of analysis of natural language, from syntax to semantics and logical representation.

References

[Ash11] Nicholas Asher. *Lexical Meaning in Context: a Web of Words*. Cambridge University Press, March 2011.

[Jac01] Evelyne Jacquey. *Ambiguïtés lexicales et traitement automatique des langues : modélisation de la polysémie logique et application aux déverbaux d'action ambigus en français*. PhD thesis, Université de Nancy 2, 2001.

[Mer15] Bruno Mery. Lexical Semantics with Linear Types. In *NLCS '15, the Third Workshop on Natural Language and Computer Science*, Kyoto, Japan, July 2015.

[MMR15] Bruno Mery, Richard Moot, and Christian Retoré. Computing the Semantics of Plurals and Massive Entities using Many-Sorted Types. In *To appear in the selected proceedings of LENLS 11*, Lecture Notes in Computer Science. Springer, 2015.

[Pus95] James Pustejovsky. *The Generative Lexicon*. MIT Press, 1995.

[RCR13] L.-M. Real-Coelho and C. Retoré. A generative montagovian lexicon for polysemous deverbal nouns. In *4th World Congress and School on Universal Logic – Workshop on Logic and Linguistics*, Rio de Janeiro, 2013.

[Ret14] Christian Retoré. The Montagovian Generative Lexicon Lambda Ty_n: a Type Theoretical Framework for Natural Language Semantics. In Ralph Matthes and Aleksy Schubert, editors, *19th International Conference on Types for Proofs and Programs (TYPES 2013)*, volume 26 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 202–229, Dagstuhl, Germany, 2014. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Calculating Projections via Type Checking ⋆

Daisuke Bekki[1,2,3] and Miho Satoh[1]

[1] Ochanomizu University, Graduate School of Humanities and Sciences ⋆⋆
[2] National Institute of Informatics ⋆⋆⋆
[3] CREST, Japan Science and Technology Agency †

1 Projection in Dependent Type Semantics

In recent years, formal semantics in the context of dependent type theory [9], which originates in Sundholm's [15] and Ranta's [12] seminal works, has achieved gradual progress: Cooper's type theory with records [4], Luo and Asher's type theory with coercive subtyping [7][1], and Martin and Pollard's [8] dynamic categorial grammar, among others. Meanwhile, dependent type semantics (DTS [2][3]) is a compositional framework of natural language semantics whose calculation of projective contents (namely, presupposition, anaphora, and conventional implicatures) reduces to *type checking* in dependent type theory, and whose presupposition binding/anaphora resolution reduces to a proof search (along the lines of Krahmer and Piwek [5][11] and Mineshima [10]). For example, in (1), for each pair of sentences, the left side of ⇒ has the right side of ⇒ as its projective content, and this empirical relation is calculated by type checking in DTS.[4]

(1) a. Sweden does not cherish its king.
⇒ Sweden has a king.

b. If Sweden is a monarchy, Sweden cherishes its king.
⇒ If Sweden is a monarchy, Sweden has a king.

c. Every monarchy cherishes its king.
⇒ Every monarchy has a king.

d. Sweden, a monarchy, cherishes its king.
⇒ Sweden is a monarchy/has a king.

⋆ Our sincere thanks to Kenichi Asai, Koji Mineshima, Ribeka Tanaka for many helpful discussions. I also thank the anonymous reviewers of TYTLES for their insightful comments. Daisuke Bekki is partially supported by JST, CREST.
⋆⋆ 2-1-1 Ohtsuka, Bunkyo-ku, Tokyo 112-8610, Japan.
⋆⋆⋆ 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, 101-8430, Japan.
† 4-1-8 Honcho, Kawaguchi, Saitama, 332-0012, Japan.
[4] Due to the page limit, the following intriguing issues are not discussed in the abstract: the intermediate accommodation case in (1c), and comparison between our "proof theoretic" account and pragmatic accounts of projection, such as the ones in Simons et al. [13] and Tonhauser et al. [16], among others.

Two concepts in DTS that enable the unity are *underspecified terms* and *local contexts*. An underspecified term (notated as $@_i$, where i is a natural number) is a proof term for a projective content. A local context is a proof term of the preceding discourse for a sentence whose representation in DTS is a function that takes a local context and returns a type (that is, a data type for proof terms). Most underspecified terms are functions that receive a local context as their argument in the context-passing mechanism of DTS. Then, anaphora resolution and presupposition binding are formulated as a substitution of underspecified terms by some proof terms, which are to be found via proof search that is unified with the inference system for calculating entailments. Different proof terms correspond to different choices of antecedents.

The whole system is integrated within a standard Montagovian compositional setting; for instance, the sentences in (1) are derived by the lexical items listed in Figure 1[5], where presupposition/CI triggers and anaphora introduce an underspecified term[6,7], which yields the semantic representations in (2).

An underspecified term of type A in DTS requires the existence of a proof term of type A, which means that A (as a proposition) is *true* regardless of the truth of the sentence itself; thus, A is projective.

Note that local contexts received by underspecified terms @ in the semantic representations in (2) vary according to their syntactic configurations. In (2a), $@_1$ receives the proof term c of its preceding discourse alone. In (2b), the proof term u of the antecedent part of the implication is also passed to $@_1$. In (2c), a given proof term is further abstracted. In (2d), $@_3$, which corresponds to the CI content, does not receive any local context.

Thus, the type of an underspecified term depends on the type of the local context it receives, but this information is recoverable by type checking. This is the reason that projective contents can be calculated via type checking in DTS.

2 Type Checking/Inference Algorithm

Despite the above, the type checking/inference algorithm for DTS remains undefined in [2]. This is due to two technical problems: 1) the undecidability of type checking in dependent type theory [14], which DTS is based on, and 2) the existence of underspecified terms.

---

[5] We adopt the notation $\begin{bmatrix} x{:}A \\ B(x) \end{bmatrix}$ for the dependent sum type $\Sigma x : A.B(x)$, and write $(x : A) \to B(x)$ for the dependent functional type $\Pi x : A.B$.

[6] The representation of the word *its* contains two underspecified terms $@_i$ and $@_j$: the former (i.e. an anaphoric part) takes a local context and returns a pair of some entity and a proof of its non-humanness. The latter (i.e. possessive presupposition) takes a local context and returns a triple of some entity, a proof that it belongs to the nominal category specified by the preceding common noun, and a proof that it is owned by *it*.

[7] The compositional analysis of the appositive in (2d) is due to Bekki and McCready [3].

PF	Syntactic categories	Semantic representations
if	$S/S/S$	$\lambda p.\lambda q.\lambda c.\,(u{:}pc) \to (q(c,u))$
every$_{nom}$	$S/(S\backslash NP)/N$	$\lambda n.\lambda p.\lambda c.\,(x{:}\mathbf{entity}) \to (u{:}nx(c,x)) \to (px(c,(x,u)))$
a$_{acc}$	$(S\backslash NP)\backslash(S\backslash NP/NP)/N$	$\lambda n.\lambda p.\lambda x.\lambda c.\left[\begin{array}{l} y{:}\mathbf{entity} \\ \left[\begin{array}{l} v{:}ny(c,y) \\ pyx(c,(y,v)) \end{array}\right] \end{array}\right]$
king	N	$\lambda x.\lambda c.\mathbf{king}(x)$
cherish	$S\backslash NP/NP$	$\lambda y.\lambda x.\lambda c.\mathbf{cherish}(x,y)$
its$_{acc}$	$(S\backslash NP)\backslash(S\backslash NP/NP)/N$	$\lambda n.\lambda p.\lambda x.\lambda c.p\left(\pi_1\left(\begin{array}{l} @_j c :: \left[\begin{array}{l} y{:}\mathbf{entity} \\ \left[\begin{array}{l} nyc \\ \mathbf{have}\left(\pi_1\left(@_i c :: \left[\begin{array}{l} z{:}\mathbf{entity} \\ \neg\mathbf{human}(z) \end{array}\right]\right), y\right) \end{array}\right] \end{array}\right] \end{array}\right)\right)xc$

Fig. 1. Lexical items in DTS (where $@_i$ and $@_j$ are underspecified terms)

(2)

a. $\lambda c.\neg\mathbf{cherish}\left(sweden, \pi_1\left(@_1(c) :: \left[\begin{array}{l} y{:}\mathbf{entity} \\ \left[\begin{array}{l} \mathbf{king}(y) \\ \mathbf{have}\left(\pi_1\left(@_2(c) :: \left[\begin{array}{l} z{:}\mathbf{entity} \\ \neg\mathbf{human}(z) \end{array}\right]\right), y\right) \end{array}\right] \end{array}\right]\right)\right)$

b. $\lambda c.(u : \mathbf{monarchy}(sweden)) \to \mathbf{cherish}\left(sweden, \pi_1\left(@_1(c,u) :: \left[\begin{array}{l} y{:}\mathbf{entity} \\ \left[\begin{array}{l} \mathbf{king}(y) \\ \mathbf{have}\left(\pi_1\left(@_2(c,u) :: \left[\begin{array}{l} z{:}\mathbf{entity} \\ \neg\mathbf{human}(z) \end{array}\right]\right), y\right) \end{array}\right] \end{array}\right]\right)\right)$

c. $\lambda c.(x : \mathbf{entity}) \to (u : \mathbf{monarchy}(x)) \to \mathbf{cherish}\left(x, \pi_1\left(@_1(c,(x,u)) :: \left[\begin{array}{l} y{:}\mathbf{entity} \\ \left[\begin{array}{l} \mathbf{king}(y) \\ \mathbf{have}\left(\pi_1\left(@_2(c,(x,u)) :: \left[\begin{array}{l} z{:}\mathbf{entity} \\ \neg\mathbf{human}(z) \end{array}\right]\right), y\right) \end{array}\right] \end{array}\right]\right)\right)$

d. $\lambda c.\left[\begin{array}{l} \mathbf{cherish}\left(sweden, \pi_1\left(@_1(c) :: \left[\begin{array}{l} y{:}\mathbf{entity} \\ \left[\begin{array}{l} \mathbf{king}(y) \\ \mathbf{have}\left(\pi_1\left(@_2(c) :: \left[\begin{array}{l} z{:}\mathbf{entity} \\ \neg\mathbf{human}(z) \end{array}\right]\right), y\right) \end{array}\right] \end{array}\right]\right)\right) \\ @_3 =_{\mathbf{monarchy}(sweden)} @_3 \end{array}\right]$

Fig. 2. Semantic representations for (1)

With regard to 1), the type checking system of the proof assistant Agda[8] employs the *annotation* construction of the form $M :: A$, which behaves just like the term M except that its type is specified as A (which turns a checkable term M into an inferable term), and presents checkable and inferable fragments of dependent type theory [6]. We adopt this strategy as well.

With regard to 2), what we need for DTS is not a type checker in the genuine sense of a functional programming language, but, instead, a checker that also *infers* a type for an underspecified term that appears in a given term. For this purpose, the type A of an underspecified term @ should be calculated from its surroundings, and when the algorithm tries to *check* whether @ has a type A within the given surroundings, it is supposed to update the list of judgements for underspecified terms by adding $\Gamma \vdash @ : A$.

However, the algorithm in [6] requires that the function in a function application construction must be an inferable term, which means that the type of an underspecified term @, in the form of function, must be inferable from the structure of @ itself. This is not the case.

[8] http://wiki.portal.chalmers.se/agda/

$$v ::= n \mid \mathsf{type} \mid \mathsf{kind} \mid \top \mid \bot \mid () \mid (x{:}v) \to v' \mid \begin{bmatrix} x{:}v \\ v' \end{bmatrix} \mid \lambda x.v \mid (v, v') \mid v \to v' \mid \begin{bmatrix} v \\ v' \end{bmatrix}$$
$$n ::= x \mid c \mid @_i \mid nv \mid \pi_i n$$
$$M_\uparrow ::= x \mid c \mid \mathsf{type} \mid (x{:}M_\downarrow) \to M_\downarrow \mid M_\uparrow M_\downarrow \mid \begin{bmatrix} x{:}M_\downarrow \\ M_\downarrow \end{bmatrix} \mid (M_\uparrow, M_\uparrow) \mid \pi_i M_\uparrow \mid M_\downarrow :: M_\downarrow \mid M_\downarrow \to M_\downarrow \mid \begin{bmatrix} M_\downarrow \\ M_\downarrow \end{bmatrix} \mid () \mid \top \mid \bot$$
$$M_\downarrow ::= M_\uparrow \mid \lambda x.M_\downarrow \mid M_\downarrow M_\uparrow \mid (M_\downarrow, M_\downarrow) \mid @_i$$

Fig. 3. Values, neutral terms, inferable and checkable terms

(3) a. (for (2a)(2d)) $\delta : \mathsf{type},\ c : \delta \vdash @_2 : \delta \to \begin{bmatrix} z{:}\mathbf{entity} \\ \neg\mathbf{human}(z) \end{bmatrix}$

b. (for (2b)) $\delta : \mathsf{type},\ c : \delta,\ u : \mathbf{monarchy}(sweden) \vdash @_2 : \begin{bmatrix} \delta \\ \mathbf{monarchy}(sweden) \end{bmatrix} \to \begin{bmatrix} z{:}\mathbf{entity} \\ \neg\mathbf{human}(z) \end{bmatrix}$

c. (for (2c)) $\delta : \mathsf{type},\ c : \delta,\ x : \mathbf{entity},\ u : \mathbf{monarchy}(x) \vdash @_2 : \begin{bmatrix} \delta \\ \begin{bmatrix} \mathbf{entity} \\ \mathbf{monarchy}(x) \end{bmatrix} \end{bmatrix} \to \begin{bmatrix} z{:}\mathbf{entity} \\ \neg\mathbf{human}(z) \end{bmatrix}$

(4) a. (for (2a)(2d)) $\delta : \mathsf{type},\ c : \delta \vdash @_1 : \delta \to \begin{bmatrix} y{:}\mathbf{entity} \\ \begin{bmatrix} \mathbf{king}(y) \\ \mathbf{have}(sweden, y) \end{bmatrix} \end{bmatrix}$

b. (for (2b)) $\delta : \mathsf{type},\ c : \delta \vdash @_1 : \begin{bmatrix} \delta \\ \mathbf{monarchy}(sweden) \end{bmatrix} \to \begin{bmatrix} y{:}\mathbf{entity} \\ \begin{bmatrix} \mathbf{king}(y) \\ \mathbf{have}(sweden, y) \end{bmatrix} \end{bmatrix}$

c. (for (2c) $\delta : \mathsf{type},\ c : \delta,\ x : \mathbf{entity},\ u : \mathbf{monarchy}(x) \vdash @_1 : \begin{bmatrix} \delta \\ \begin{bmatrix} \mathbf{entity} \\ \mathbf{monarchy}(x) \end{bmatrix} \end{bmatrix} \to \begin{bmatrix} y{:}\mathbf{entity} \\ \begin{bmatrix} \mathbf{king}(y) \\ \mathbf{have}(x, y) \end{bmatrix} \end{bmatrix}$

(5) a. (for (2d)) $\delta : \mathsf{type},\ c : \delta \vdash @_3 : \mathbf{monarchy}(sweden)$

Fig. 4. Projective contents of (2)

The key observation to solve the second problem is that an underspecified term @ in DTS is always a simply typed function. Therefore, it suffices for the type of a simply typed function @ to be inferable from the type of a local context c and the type of the application $@(c)$ (the same situation arises for simply typed pairs, but we do not discuss the details of that case here). The main contributions of this work are as follows.

(i) We extend the fragment of dependent type theory given in [6] to include underspecified terms and dependent sum types, as defined in Figure 3, where $v, n, M_\uparrow$ and $M_\downarrow$ are the collections of *values*, *neutral terms*, *inferable terms*, and *checkable terms*, respectively.
(ii) We extend the type checking/inference rules of [6] as in Figure 5 for the fragment in Figure 3, where simply typed functions (and simply typed pairs) are inferable terms.

Note that any annotations that are required for the representation language of DTS to be confined to the fragment in Figure 3 can be naturally specified within the lexical representations of presupposition triggers, as shown in Figure 1.

Note also that the soundness of this algorithm—namely, that every judgment that is checked as true or inferred in this algorithm is also derivable in the original

$$\frac{[L]\ \Gamma \vdash_\sigma M :\uparrow v\ [L']}{[L]\ \Gamma \vdash_\sigma M :\downarrow v\ [L']}\ (CHK) \quad \frac{(x,v) \in \Gamma}{[L]\ \Gamma \vdash_\sigma x :\uparrow v\ [L]}\ (VAR) \quad \frac{(c,v) \in \sigma}{[L]\ \Gamma \vdash_\sigma c :\uparrow v\ [L]}\ (CON) \quad \frac{}{[L]\ \Gamma \vdash_\sigma \mathsf{type} :\uparrow \mathsf{kind}\ [L]}\ (TYPE)$$

$$\frac{[L]\ \Gamma \vdash_\sigma A :\downarrow \mathsf{s}_1\ [L'] \quad A \Downarrow_\beta v \quad [L']\ \Gamma,\ x : v \vdash_\sigma B :\downarrow \mathsf{s}_2\ [L'']}{[L]\ \Gamma \vdash_\sigma (x{:}A) \to B :\uparrow \mathsf{s}_2\ [L'']}\ (\Pi F) \quad (\mathsf{s}_1, \mathsf{s}_2 \in \{\mathsf{type}, \mathsf{kind}\})$$

$$\frac{[L]\ \Gamma,\ x : v \vdash_\sigma M :\downarrow v'\ [L']}{[L]\ \Gamma \vdash_\sigma \lambda x.M :\downarrow (x{:}v) \to v'\ [L']}\ (\Pi I) \quad \frac{[L]\ \Gamma \vdash_\sigma M :\uparrow (x{:}v) \to v'\ [L'] \quad [L']\ \Gamma \vdash_\sigma N :\downarrow v\ [L''] \quad v'[N/x] \Downarrow_\beta v''}{[L]\ \Gamma \vdash_\sigma MN :\uparrow v''\ [L'']}\ (\Pi E)$$

$$\frac{[L]\ \Gamma \vdash_\sigma A :\downarrow \mathsf{s}_1\ [L'] \quad [L']\ \Gamma \vdash_\sigma B :\downarrow \mathsf{s}_2\ [L'']}{[L]\ \Gamma \vdash_\sigma A \to B :\uparrow \mathsf{s}_2\ [L'']}\ (\to F) \quad (\mathsf{s}_1, \mathsf{s}_2 \in \{\mathsf{type}, \mathsf{kind}\})$$

$$\frac{[L]\ \Gamma,\ x : v \vdash_\sigma M :\downarrow v'\ [L']}{[L]\ \Gamma \vdash_\sigma \lambda x.M :\downarrow v \to v'\ [L']}\ (\to I) \quad \frac{[L]\ \Gamma \vdash_\sigma N :\uparrow v\ [L'] \quad [L']\ \Gamma \vdash_\sigma M :\downarrow v \to v'\ [L'']}{[L]\ \Gamma \vdash_\sigma MN :\downarrow v'\ [L'']}\ (\to E)$$

$$\frac{[L]\ \Gamma \vdash_\sigma A :\downarrow \mathsf{s}_1\ [L'] \quad A \Downarrow_\beta v \quad [L']\ \Gamma,\ x : v \vdash_\sigma B :\downarrow \mathsf{s}_2\ [L'']}{[L]\ \Gamma \vdash_\sigma \begin{bmatrix} x{:}A \\ B \end{bmatrix} :\uparrow \mathsf{s}_2\ [L'']}\ (\Sigma F) \quad (\mathsf{s}_1, \mathsf{s}_2 \in \{\mathsf{type}, \mathsf{kind}\})$$

$$\frac{[L]\ \Gamma \vdash_\sigma M :\downarrow v\ [L'] \quad v'[M/x] \Downarrow_\beta v'' \quad [L']\ \Gamma \vdash_\sigma N :\downarrow v''\ [L'']}{[L]\ \Gamma \vdash_\sigma (M,N) :\downarrow \begin{bmatrix} x{:}v \\ v' \end{bmatrix}\ [L'']}\ (\Sigma I)$$

$$\frac{[L]\ \Gamma \vdash_\sigma M :\uparrow \begin{bmatrix} x{:}v \\ v' \end{bmatrix}\ [L']}{[L]\ \Gamma \vdash_\sigma \pi_1 M :\uparrow v\ [L']}\ (\Sigma E) \quad \frac{[L]\ \Gamma \vdash_\sigma M :\uparrow \begin{bmatrix} x{:}v \\ v' \end{bmatrix}\ [L'] \quad v'[\pi_1 M/x] \Downarrow_\beta v''}{[L]\ \Gamma \vdash_\sigma \pi_2 M :\uparrow v''\ [L']}\ (\Sigma E)$$

$$\frac{[L]\ \Gamma \vdash_\sigma A :\downarrow \mathsf{s}_1\ [L'] \quad [L']\ \Gamma \vdash_\sigma B :\downarrow \mathsf{s}_2\ [L'']}{[L]\ \Gamma \vdash_\sigma \begin{bmatrix} x{:}A \\ B \end{bmatrix} :\uparrow \mathsf{s}_2\ [L'']}\ (\to F) \quad (\mathsf{s}_1, \mathsf{s}_2 \in \{\mathsf{type}, \mathsf{kind}\})$$

$$\frac{[L]\ \Gamma \vdash_\sigma M :\uparrow v\ [L'] \quad [L']\ \Gamma \vdash_\sigma N :\uparrow v'\ [L'']}{[L]\ \Gamma \vdash_\sigma (M,N) :\uparrow \begin{bmatrix} v \\ v' \end{bmatrix}\ [L'']}\ (\wedge I) \quad \frac{[L]\ \Gamma \vdash_\sigma M :\uparrow \begin{bmatrix} v \\ v' \end{bmatrix}\ [L']}{[L]\ \Gamma \vdash_\sigma \pi_1 M :\uparrow v\ [L']}\ (\wedge E) \quad \frac{[L]\ \Gamma \vdash_\sigma M :\uparrow \begin{bmatrix} v \\ v' \end{bmatrix}\ [L']}{[L]\ \Gamma \vdash_\sigma \pi_2 M :\uparrow v'\ [L']}\ (\wedge E)$$

$$\frac{[L]\ \Gamma \vdash_\sigma A :\downarrow \mathsf{s}\ [L'] \quad A \Downarrow_\beta v \quad [L']\ \Gamma \vdash_\sigma M :\downarrow v\ [L'']}{[L]\ \Gamma \vdash_\sigma M :: A :\uparrow v\ [L'']}\ (ANN) \quad (\mathsf{s} \in \{\mathsf{type}, \mathsf{kind}\})$$

$$\frac{}{[L]\ \Gamma \vdash_\sigma \top :\uparrow \mathsf{type}\ [L]}\ (\top F) \quad \frac{}{[L]\ \Gamma \vdash_\sigma () :\uparrow \top\ [L]}\ (\top I) \quad \frac{}{[L]\ \Gamma \vdash_\sigma \bot :\uparrow \mathsf{type}\ [L]}\ (\bot F)$$

$$\frac{}{[L]\ \Gamma \vdash_\sigma @_i :\downarrow v\ [L, (\Gamma \vdash_\sigma @_i : v)]}\ (ASP)$$

Fig. 5. Type checking/inference rules for DTS

typing rule of dependent type theory—should be proven by a straightforward induction on constructions.

3 Implementation

We implemented the algorithm in Figure 5, using the functional programming language Haskell. The program includes two main functions: `typeInfer` and `typeCheck`. The function `typeInfer` 1) takes a global context Γ and a term M, and returns a type A such that $\Gamma \vdash M : A$; and 2) updates the current list $[L]$ of judgments for underspecified terms. The function `typeCheck` 1) takes a global context Γ, a term M, and a type A, and returns a Boolean value indicating

whether $\Gamma \vdash M : A$ holds; and 2) updates the current list $[L]$ of judgments for underspecified terms.

4 Calculating Projective Contents

Given terms (2) as inputs, our type checking program checks whether it has a type $\delta \to \mathsf{type}$ within a global context $\delta : \mathsf{type},\ c : \delta$ (δ is a propositional content of its preceding discourse and c is its proof term), and updates the list of judgments for underspecified terms so as to contain those in Figure 4. The lists (3) and (4) are, respectively, for $@_2$ and $@_1$, which correspond to the anaphoric part and the possessive presupposition part of *its*. The list (5) is for the appositive CI in (2d).

The judgments in (4) are the ones obtained after anaphora resolution for $@_2$ has been executed: the output of the program contains the occurrence of $@_2$ within the types for $@_1$ (this means that the possessive presuppositions in (1) contain the anaphora antecedents). The proof terms corresponding to the intended reading in (1) (i.e. *it* refers to *Sweden* in (1a)(1b)(1d), and *every monarchy* in (1c)) are as follows:

$$@_2 = \lambda c.(\mathit{sweden}, n(\mathit{sweden})m)$$
$$@_2 = \lambda c.(\pi_1\pi_2(c), n(\pi_1\pi_2(c))(\pi_2\pi_2(c))),$$

where m is a proof term for $\mathbf{monarchy}(\mathit{sweden})$ (*Sweden is a monarchy*), and n is a proof term for $(x : \mathbf{entity}) \to \mathbf{monarchy}(x) \to \neg\mathbf{human}(x)$ (*Monarchy is not a human*). Then, (4) is obtained by substituting $@_2$ with these proof terms.

The judgments for $@_1$ in (4) correctly represent the projective contents in (1): the type of (4a) states that the preceding discourse entails that Sweden has a king (which is a projective content of (1a)). The type of (4b) states that the preceding discourse entails that, if Sweden is a monarchy, then Sweden has a king. (This is a projective content of (1b). The type of (4c) states that the preceding discourse entails that every monarchy has a king (which is a projective content of (1c).) The type of (5) is the projected CI content in (1d), stating that Sweden is a monarchy.

5 Conclusion

We show that the projective contents of (1) are automatically calculated from the semantic representations via the type checking/inference algorithm in Figure 5. In future work, we plan to connect the input of this program to a robust CCG parser, or, independently, to connect the output of this program to a theorem prover, aiming at establishing a pipeline from sentences to their entailment relation, including projective contents such as presuppositions, anaphora and conventional implicatures.

References

1. Asher, N., Luo, Z.: Formalisation of coercions in lexical semantics. In: Sinn und Bedeutung 17. pp. 63–80. Paris (2012)
2. Bekki, D.: Representing anaphora with dependent types. In: Asher, N., Soloviev, S.V. (eds.) Logical Aspects of Computational Linguistics (8th international conference, LACL2014, Toulouse, France, June 2014 Proceedings), LNCS 8535. pp. 14–29. Springer, Heiderburg, Toulouse (2014)
3. Bekki, D., McCready, E.: Ci via dts. In: LENLS11. pp. 110–123. Tokyo (2014)
4. Cooper, R.: Austinian truth, attitudes and type theory. Research on Language and Computation 3, 333–362 (2005)
5. Krahmer, E., Piwek, P.: Presupposition projection as proof construction. In: Bunt, H., Muskens, R. (eds.) Computing Meanings: Current Issues in Computational Semantics. Studies in Linguistics Philosophy Series, Kluwer Academic Publishers, Dordrecht (1999)
6. Löh, A., McBride, C., Swierstra, W.: A tutorial implementation of a dependently typed lambda calculus. Fundamenta Informaticae - Dependently Typed Programming 102(2), 177–207 (2010)
7. Luo, Z.: Common nouns as types. In: Béchet, D., Dikovsky, A. (eds.) Logical Aspects of Computational Linguistics, 7th International Conference, LACL2012, Nantes, France, July 2012 Proceedings, pp. 173–185. Springer (2012)
8. Martin, S., Pollard, C.J.: A dynamic categorial grammar. In: Formal Grammar 19, LNCS 8612 (2014)
9. Martin-Löf, P.: Intuitionistic Type Theory, vol. 17. Italy: Bibliopolis, Naples (1984), sambin, Giovanni (ed.)
10. Mineshima, K.: A presuppositional analysis of definite descriptions in proof theory. In: Satoh, K., Inokuchi, A., Nagao, K., Kawamura, T. (eds.) JSAI 2007 LNAI, vol. 4914, pp. 214–227. Springer, Heidelberg (2008)
11. Piwek, P., Krahmer, E.: Presuppositions in context: Constructing bridges. In: Bonzon, P., Cavalcanti, M., Nossum, R. (eds.) Formal Aspects of Context. Applied Logic Series, Kluwer Academic Publishers, Dordrecht (2000)
12. Ranta, A.: Type-Theoretical Grammar. Oxford University Press (1994)
13. Simons, M., Tonhauser, J., Beaver, D.I., Roberts, C.: What projects and why. In: SALT 20. pp. 309–327 (2010)
14. Sørensen, M., Urzyczyn, P.: Lectures on the Curry–Howard Isomorphism. Studies in Logic and the Foundation of Mathematics, Elsevier (2006)
15. Sundholm, G.: Proof theory and meaning. In: Gabbay, D., Guenthner, F. (eds.) Handbook of Philosophical Logic, vol. III, pp. 471–506. Kluwer, Reidel (1986)
16. Tonhauser, J., Beaver, D.I., Roberts, C., Simons, M.: Towards a taxonomy of projective content. Language 89(1) (2013)

Quantification in Frame Semantics with Hybrid Logic*

Laura Kallmeyer[1], Timm Lichte[1], Rainer Osswald[1], Sylvain Pogodalla[1,2], and Christian Wurm[1]

[1] Heinrich Heine Universität, Düsseldorf
{laura.kallmeyer,timm.lichte,rainer.osswald,christian.wurm}
@phil.uni-duesseldorf.de
[2] INRIA, Villers-lès-Nancy, F-54600, France
Université de Lorraine, LORIA, UMR 7503, Vandœuvre-lès-Nancy, F-54500, France
CNRS, LORIA, UMR 7503, Vandœuvre-lès-Nancy, F-54500, France
sylvain.pogodalla@inria.fr

Abstract. This paper aims at integrating logical operators into frame-based semantics. Frames are semantic graphs that allow to capture lexical meaning in a fine-grained way but that do not come with a natural way to integrate logical operators such as quantifiers. The approach we propose starts from the observation that modal logic is a powerful tool for describing relational structures, hence frames. We use its hybrid logic extension in order to incorporate quantification and thereby allow for inference and reasoning. We develop a type theoretic compositional semantics using this approach, formulated within Abstract Categorial Grammar.

1 Frames and Lexical Semantics

Frames emerged as a representation format of conceptual and lexical knowledge [10,4,15]. They are commonly presented as semantic graphs with labelled nodes and edges, such as the one in Fig. 1, where nodes correspond to entities (individuals, events, ...) and edges correspond to (functional or non-functional) relations between these entities. In Fig. 1 all relations except *part-of* are meant to be functional. Frames can be formalized as extended typed feature structures [18,12,14], but a reformulation in first order logic is also straightforward [12]. This conception of frames is therefore not to be confused with the somewhat simpler FrameNet frames (see [17]).

Recent work has addressed the composition of lexical frames on the sentential level by means of an explicit syntax-semantics interface [12]. However, the integration of logical operators remains a desideratum. While [13] presents an experiment with a seamless intergration of "quantifier frames", [16] suggests to keep frames and logical operators separate. We follow the latter general approach in this paper.

* This work was supported by the INRIA sabbatical program and by the CRC 991 "The Structure of Representations in Language, Cognition, and Science" funded by the German Research Foundation (DFG).

2 Hybrid Logic and Semantic Frames

2.1 Hybrid Logic

With the notations of [2], Rel is a set of relational symbols, Prop a set of propositional variables, Nom a set of nominals, and Svar a set of state variables ($\mathsf{Stat} = \mathsf{Nom} \cup \mathsf{Svar}$). The language of formulas is $\mathsf{Forms} ::= \top \mid p \mid s \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \langle R\rangle\phi \mid \mathbf{\exists}\phi \mid @_s\phi \mid \downarrow x.\phi \mid \exists x.\phi$ where $p \in \mathsf{Prop}$, $s \in \mathsf{Stat}$, $R \in \mathsf{Rel}$ and $\phi, \phi_1, \phi_2 \in \mathsf{Forms}$. A *model* $\mathcal{M}$ is a triple $\langle M, (R^{\mathcal{M}})_{R\in\mathsf{Rel}}, V\rangle$ such that M is a non-empty set, each $R^{\mathcal{M}}$ is a binary relation on M, and the valuation $V : \mathsf{Prop} \cup \mathsf{Nom} \longrightarrow \wp(M)$ is such that if $i \in \mathsf{Nom}$ then $V(i)$ is a singleton. An assignment g is a mapping $g : \mathsf{Svar} \longrightarrow M$. For an assignment g, g^x_m is an assignment that differs from g at most on x and $g^x_m(x) = m$. For $s \in \mathsf{Stat}$, we also define $[s]^{\mathcal{M},g}$ to be the only m such that $V(s) = \{m\}$ if $s \in \mathsf{Nom}$ and $[s]^{\mathcal{M},g} = g(s)$ if $s \in \mathsf{Svar}$.

Let $\mathcal{M}$ be a model, $w \in M$, and g an assignment for $\mathcal{M}$. The *satisfaction relation* is defined as follows:

$\mathcal{M}, g, w \vDash \top$
$\mathcal{M}, g, w \vDash s$ iff $w = [s]^{\mathcal{M},g}$ for $s \in \mathsf{Stat}$
$\mathcal{M}, g, w \vDash \neg\phi$ iff $\mathcal{M}, g, w \nvDash \phi$
$\mathcal{M}, g, w \vDash \phi_1 \wedge \phi_2$ iff $\mathcal{M}, g, w \vDash \phi_1$ and $\mathcal{M}, g, w \vDash \phi_2$
$\mathcal{M}, g, w \vDash \langle R\rangle\phi$ iff there is a $w' \in M$ such that $R^{\mathcal{M}}(w, w')$ and $\mathcal{M}, g, w' \vDash \phi$
$\mathcal{M}, g, w \vDash p$ iff $w \in V(p)$ for $p \in \mathsf{Prop}$
$\mathcal{M}, g, w \vDash @_s\phi$ iff $\mathcal{M}, g, [s]^{\mathcal{M},g} \vDash \phi$ for $s \in \mathsf{Stat}$
$\mathcal{M}, g, w \vDash \downarrow x.\phi$ iff $\mathcal{M}, g^x_w, w \vDash \phi$
$\mathcal{M}, g, w \vDash \exists x.\phi$ iff there is a $w' \in M$ such that $\mathcal{M}, g^x_{w'}, w \vDash \phi$
$\mathcal{M}, g, w \vDash \mathbf{\exists}\phi$ iff there is a $w' \in M$ such that $\mathcal{M}, g, w' \vDash \phi$

We also define $\mathbf{\forall}\phi \equiv \neg\mathbf{\exists}(\neg\phi)$ (i.e., $\mathcal{M}, g, w \vDash \mathbf{\forall}\phi$ iff $\forall w'\ \mathcal{M}, g, w' \vDash \phi$)[3] and $\phi \Longrightarrow \psi \equiv (\neg\phi) \vee \psi$. A formula ϕ is:

- *satisfiable* if there is a model $\mathcal{M}$, and assignment g on $\mathcal{M}$, and a state $w \in M$ such that $\mathcal{M}, g, w \vDash \phi$
- *globally true* in a model $\mathcal{M}$ under an assignment g, i.e., $\mathcal{M}, g, w \vDash \phi$ for all $w \in M$. We write $\mathcal{M}, g \vDash \phi$

2.2 Feature Structures

In [12], semantic frames are introduced as *base-labelled feature structure with types and relations*. This definition extends the standard definition of feature structures in two respects: In addition to features, proper relations between nodes can be expressed. Moreover, it is not required that every node is accessible from

[3] According to the satisfaction relation, $\downarrow$ and $\exists$ bind state variables without changing the current evaluation state. [7] shows that they define a distinct hierarchy from the one we get using $\mathbf{\exists}$ (or some other binder Σ). It also shows that the fragment using operators from the two hierarchies is as expressive as the most expressive fragment.

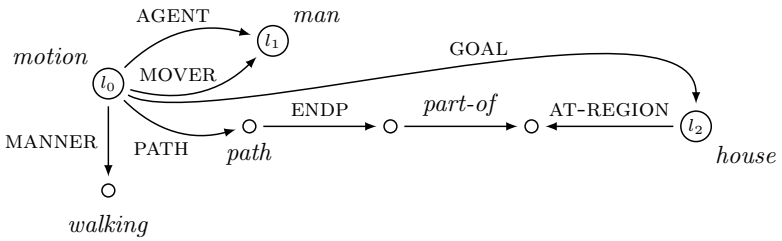


Fig. 1. Frame for the meaning of *the man walked to the house* (adapted from [12])

a single root node via a feature path; instead, it is required that every node is accessible from one of the base-labelled nodes. Semantic frames defined in this way can be seen as finite first-order structures which conform to a signature consisting of a set $\mathsf{Label} \cup \mathsf{Type}$ of unary relation symbols and a set $\mathsf{Feat} \cup \mathsf{Rel}$ of binary relation symbols subject to the constraints that the members of Label denote singletons, the members of Feat denote *functional* relations, and that the above accessibility condition holds. In the example frame of Fig. 1, symbols inside nodes (l_0, l_1, ...) indicate base labels, symbols attached to nodes (*man*, *motion*, ...) belong to Type, members of Feat are marked by small caps (AGENT, ENDP, ...), and *part-of* is the only member of Rel occurring in the frame.

Structures of this kind can easily be turned into Kripke structures by treating the interpretation of the members of $\mathsf{Label} \cup \mathsf{Type}$ by a separate valuation function. Semantic frames, or feature structures, provide thus a natural application domain for modal languages and, in particular, for hybrid extensions because of the need to cope with node labels and feature path re-entrancies [5]. Under the formal set-up of Section 2.1, Type corresponds to Prop, Label corresponds to Nom, and Feat is subsumed under Rel. (The functionality of the members of Feat must be enforced separately.) The semantic frame of Fig. 1 is a model that satisfies the formula (1) at the element named by l_0.

$$(1)\quad l_0 \wedge \mathit{motion} \wedge \langle \textsc{agent} \rangle (l_1 \wedge \mathit{man}) \wedge \langle \textsc{mover} \rangle l_1 \wedge \langle \textsc{goal} \rangle (l_2 \wedge \mathit{house}) \wedge \langle \textsc{manner} \rangle \mathit{walking} \wedge (\exists v\ w . \langle \textsc{path} \rangle (\mathit{path} \wedge \langle \textsc{endp} \rangle v) \wedge @_{l_2}(\langle \textsc{at-region} \rangle w) \wedge @_v(\langle \mathit{part\text{-}of} \rangle w))$$

The logical framework of [12] does not provide means for explicit quantification. As a consequence, the referential entities of the domain of discourse are implicitly treated as definite, which is reflected by the crucial role of nominals in (1).[4] In

[4] Hybrid logic with nominals but without quantification over states also allows [3] to describe semantic dependency graphs. Natural language quantification is encoded using RESTR and BODY relations. However, it is not clear how to compute relations between these representations (e.g., how to check that *John kisses Mary* holds in case *every man kisses Mary* holds).

the following, we will show how this limitation can be overcome by employing hybrid languages.

3 Type-Theoretic Semantics with Frames

We now provide the type-theoretic syntax-semantics interface allowing for a compositional building of the meanings. We describe it using the ACG [9] framework. As we are concerned in this article with semantic modeling and quantification rather than with parsing, we use higher-order types for quantified noun-phrases.

The models we are considering are *semantic frames* instead of arbitrary first-order models. So we first present some models in which we consider the sentences (2a–4a). When the model is the frame of Fig. 2(a), we expect (2a) to be true as there is a *kissing* event with AGENT and THEME attributes linking to persons named (linked with the NAMED attribute) *John* and *Mary* resp. (3a) is expected to be false, as well as (4a) with the object wide scope reading as there is a person named *Paul* (resp. named *Peter*) who is AGENT of a single *kissing* event whose THEME is a person named *Sue* (resp. *Mary*). On the other hand, the subject wide scope reading of (4a) is expected to be true. (5a) shows how state storing with the $\downarrow$ operator correctly interacts with the @ operator used in specifying node sharing (for instance that the state linked with a GOAL relation in the verb semantic recipe has to be specified by the *PP*). This sentence is expected to be true (both readings) in the model given by the frame of Fig. 2(b).

(2) a. *John kisses Mary*

b. $\mathbf{\exists}(\mathit{kissing} \wedge \langle \text{AGENT} \rangle (\mathit{person} \wedge \langle \text{NAMED} \rangle \mathit{John}) \wedge \langle \text{THEME} \rangle (\mathit{person} \wedge \langle \text{NAMED} \rangle \mathit{Mary}))$

(3) a. *Every man kisses Mary*

b. $\mathbf{\forall}(\downarrow i.\mathit{man} \implies \mathbf{\exists}(\mathit{kissing} \wedge \langle \text{AGENT} \rangle i \wedge \langle \text{THEME} \rangle (\mathit{person} \wedge \langle \text{NAMED} \rangle \mathit{Mary})))$

(4) a. *Every man kisses some woman*

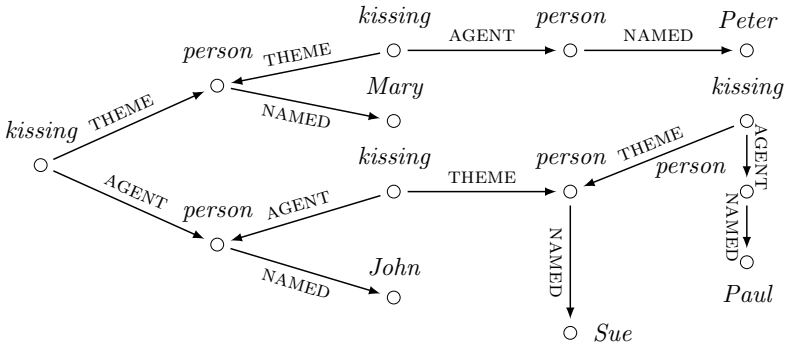
b. $\mathbf{\forall}(\downarrow i.\mathit{man} \implies \mathbf{\exists}(\downarrow i'.\mathit{woman} \wedge \mathbf{\exists}(\mathit{kissing} \wedge \langle \text{AGENT} \rangle i \wedge \langle \text{THEME} \rangle i')))$

c. $\mathbf{\exists}(\downarrow i.\mathit{woman} \wedge \mathbf{\forall}(\downarrow i'.\mathit{man} \implies \mathbf{\exists}(\mathit{kissing} \wedge \langle \text{AGENT} \rangle i' \wedge \langle \text{THEME} \rangle i)))$

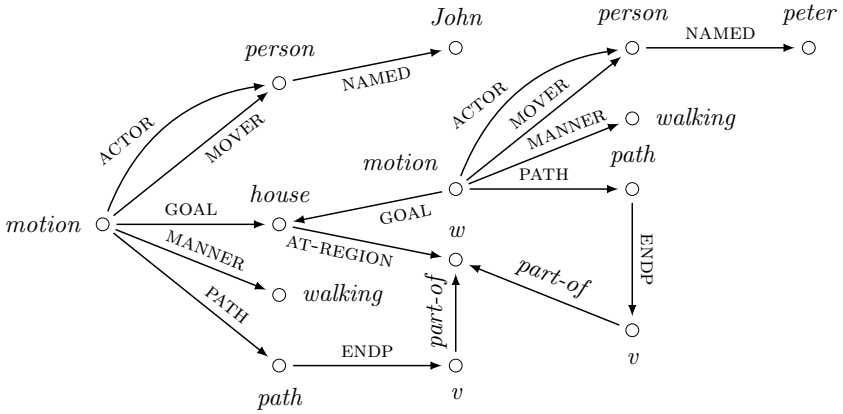
(5) a. *Every man walked to some house*

b. $\mathbf{\exists}(\downarrow i.\mathit{house} \wedge (\mathbf{\forall}(\downarrow i'.\mathit{man} \implies (\exists a\ g.\mathbf{\exists}(\mathit{motion} \wedge \langle \text{AGENT} \rangle a \wedge \langle \text{MOVER} \rangle a \wedge \langle \text{GOAL} \rangle g \wedge \langle \text{PATH} \rangle \mathit{path} \wedge \langle \text{MANNER} \rangle \mathit{walking} \wedge @_a i' \wedge (\exists r\ v\ w.\mathit{event} \wedge \langle \text{PATH} \rangle (\mathit{path} \wedge \langle \text{ENDP} \rangle v) \wedge @_r(\langle \text{AT-REGION} \rangle w) \wedge @_v(\langle \mathit{part\text{-}of} \rangle w) \wedge @_r(g \wedge i)))))))$

c. $\mathbf{\forall}(\downarrow i.\mathit{man} \implies (\mathbf{\exists}(\downarrow i'.\mathit{house} \wedge (\exists a\ g.\mathbf{\exists}(\mathit{motion} \wedge \langle \text{AGENT} \rangle a \wedge \langle \text{MOVER} \rangle a \wedge \langle \text{GOAL} \rangle g \wedge \langle \text{PATH} \rangle \mathit{path} \wedge \langle \text{MANNER} \rangle \mathit{walking} \wedge @_a i \wedge (\exists r\ v\ w.\mathit{event} \wedge \langle \text{PATH} \rangle (\mathit{path} \wedge \langle \text{ENDP} \rangle v) \wedge @_r(\langle \text{AT-REGION} \rangle w) \wedge @_v(\langle \mathit{part\text{-}of} \rangle w) \wedge @_r(g \wedge i')))))))$



(a) Quantification



(b) Quantification and node sharing

Fig. 2. Frame samples

As in [12], the syntax-semantics interface we propose builds a *frame description* out of a sentence in natural language. This frame description is a logical formula that is checked against the possible models, and the sentence is true w.r.t. a model $\mathcal{M}$ in case this model satisfies the logical formula. More precisely, given a sentence s and its semantic representation $[\![s]\!]$, we say that s *is true* iff for all assignments g, $\mathcal{M}, g \vDash [\![s]\!]$ (i.e., $[\![s]\!]$ is globally true in $\mathcal{M}$ under any assignment).

We use the following syntactic types: NP, S, N, and PP and the following syntactic type assignments:

John, Mary : NP	kisses : $NP \to NP \to S$	to, into : $NP \to PP$
man, woman, house : N	every, some : $N \to (NP \to S) \to S$	walk : $PP \to NP \to S$

$event, kissing, motion, person, John, Mary, \ldots : t$ $\quad @ : t \to t \to t$
$\wedge : t \to t \to t$ $\quad \Longrightarrow : t \to t \to t$ $\quad \mathbf{\exists}, \mathbf{\forall} : t \to t$ $\quad \downarrow, \exists : (t \to t) \to t$

Table 1. Constant terms of the semantic language

S, NP, N	$:= t$	PP	$:= t \to t$
John	$:= John$	Mary	$:= Mary$
man	$:= man$	woman	$:= woman$
house	$:= house$		
some	$:= \lambda P\ Q.\mathbf{\exists}(\downarrow i.P \wedge (Q\ i))$	every	$:= \lambda P\ Q.\mathbf{\forall}(\downarrow i.P \Longrightarrow (Q\ i))$
kisses	$:= \lambda o\ s.\mathbf{\exists}(kissing \wedge \langle \text{AGENT} \rangle s \wedge \langle \text{THEME} \rangle o)$		
walks	$:= \lambda pp\ s.\exists a\ g.\mathbf{\exists}(motion \wedge \langle \text{AGENT} \rangle a \wedge \langle \text{MOVER} \rangle a \wedge \langle \text{GOAL} \rangle g$		
	$\wedge \langle \text{PATH} \rangle path \wedge \langle \text{MANNER} \rangle walking \wedge @_a s \wedge (pp\ g))$		
to	$:= \lambda n\ g.\exists r\ v\ w.event \wedge \langle \text{PATH} \rangle (path \wedge \langle \text{ENDP} \rangle v) \wedge$		
	$@_r \langle \text{AT-REGION} \rangle w \wedge @_v \langle part\text{-}of \rangle w \wedge @_r (g \wedge n)$		
into	$:= \lambda n\ g.\exists r\ v\ w.event \wedge \langle \text{PATH} \rangle (path \wedge \langle \text{ENDP} \rangle v) \wedge$		
	$@_r \langle \text{IN-REGION} \rangle w \wedge @_v \langle part\text{-}of \rangle w \wedge @_r (g \wedge n)$		

Table 2. Semantic interpretation of constants

Table 1 shows the semantic constants we use, including logical operators and quantifiers. We follow [12] in the semantics and meaning decomposition of locomotion verbs.

Then we can use the semantic interpretation given in Table 2. For sake of conciseness and explanatory purposes, we use a single type t to denote modal formulas. This is not completely satisfactory as we can build terms that are not in Forms. (In principle, any proposition could specify the $@ : t \to t \to t$ operator. But in our lexicon example, we of course restrict the first parameter to state variables.) A more faithful encoding could use the standard parametrization of the propositions with a s type for states, or use a dedicated hybrid type-theory [1]. Then the following equalities hold, where t_{2b} is the term in (2b), t_{3b} is the term in (3b), etc.:

(6) $\quad [\![\text{kisses Mary John}]\!] = t_{2b}$

(7) $\quad [\![(\text{every man})\ (\lambda x.\text{kisses Mary } x)]\!] = t_{3b}$

(8) $\quad [\![(\text{every man})\ (\lambda x.(\text{some woman})\ (\lambda y.\text{kisses } y\ x))]\!] = t_{4b}$

(9) $\quad [\![(\text{some woman})\ (\lambda y.(\text{every man})\ (\lambda x.\text{kisses } y\ x))]\!] = t_{4c}$

In (10) and (11), we have an interaction of the storing for quantification and path equalities compositionally deriving from the verb and the preposition. In the verb semantics, the path equalities specify that the MOVER and the AGENT attributes of the event are the same, and that the information provided by the *pp* argument should hold for the GOAL g. In its semantics, the preposition contributes on the one hand to the main event (as the *event* proposition is evaluated at the current state) and on the other hand by specifying that the g state (meant to be the target node of the verb that the proposition modifies, here the target of the GOAL attribute) should be identified to the n argument

(the noun phrase which is argument of the preposition).

(10) $[\![(\text{every man})\ (\lambda x.(\text{some house})(\lambda y.\text{walked (to } y)\ x))]\!] = t_{5b}$

(11) $[\![(\text{some house})(\lambda y.(\text{every man})\ (\lambda x.\text{walked (to } y)\ x))]\!] = t_{5c}$

4 Conclusion and Perspectives

We used hybrid logic as a means to integrate logical operators with frame semantics. A type theoretic semantics was presented that shows how to compositionally derive different quantifier scope readings. This approach has much in common with [16], which combines data semantics with frame semantics. The exact relation between the two approaches needs to be spelled out in future work. We see applications of our approach of using hybrid logic for frame semantics in the context of various formalisms; we plan in particular to pursue this approach in the framework of Lexicalized Tree Adjoining Grammars (LTAG) [12].

We also think that the compositional account we presented allows us to consider an embedding within a underspecified representation language. The object language (in the sense of [8]) would be the hybrid logic language instead of the usual first-order logic language, following a standard modeling of scope ambiguity in LTAG.

Finally, we plan to investigate the computational properties of the framework we propose with respect to the hybrid inferential systems [6] and the specific properties induced by the frame models we consider, typically the functionality of the attribute relations [19].

References

1. Areces, C., Blackburn, P., Huertas, A., Manzano, M.: Completeness in hybrid type theory. Journal of Philosophical Logic 43(2-3), 209–238 (2014), DOI: 10.1007/s10992-012-9260-4
2. Areces, C., ten Cate, B.: Hybrid logics. In: Blackburn, P., Benthem, J.V., Wolter, F. (eds.) Handbook of Modal Logic, Studies in Logic and Practical Reasoning, vol. 3, chap. 14, pp. 821–868. Elsevier (2007), DOI: 10.1016/S1570-2464(07)80017-6
3. Baldridge, J., Kruijff, G.J.: Coupling CCG and hybrid logic dependency semantics. In: Proceedings of 40th Annual Meeting of the Association for Computational Linguistics. pp. 319–326. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA (July 2002), DOI: 10.3115/1073083.1073137
4. Barsalou, L.: Frames, concepts, and conceptual fields. In: Lehrer, A., Kittey, E.F. (eds.) Frames, fields, and contrasts: New essays in semantic and lexical organization, pp. 21–74. Lawrence Erlbaum Associates, Hillsdale (1992)
5. Blackburn, P.: Modal logic and attribute value structures. In: de Rijke, M. (ed.) Diamonds and Defaults, Synthese Library, vol. 229, pp. 19–65. Springer Netherlands (1993), DOI: 10.1007/978-94-015-8242-1_2
6. Blackburn, P., Marx, M.: Tableaux for quantified hybrid logic. In: Egly, U., Fermüller, C. (eds.) Automated Reasoning with Analytic Tableaux and Related Methods, Lecture Notes in Computer Science, vol. 2381, pp. 38–52. Springer Berlin Heidelberg (2002), DOI: 10.1007/3-540-45616-3_4

7. Blackburn, P., Seligman, J.: Hybrid languages. Journal of Logic, Language and Information 4(3), 251–272 (1995), DOI: 10.1007/BF01049415
8. Bos, J.: Predicate logic unplugged. In: Proceedings of the Tenth Amsterdam Colloquium (1995), http://www.let.rug.nl/bos/pubs/Bos1996AmCo.pdf
9. de Groote, P.: Towards Abstract Categorial Grammars. In: Association for Computational Linguistics, 39th Annual Meeting and 10th Conference of the European Chapter, Proceedings of the Conference. pp. 148–155 (2001), ACL anthology: P01-1033
10. Fillmore, C.J.: The case for case reopened. In: Cole, P., Sadock, J.M. (eds.) Grammatical Relations, Syntax and Semantics, vol. 8, pp. 59–81. Academic Press, New York (1977)
11. Gamerschlag, T., Gerland, D., Osswald, R., Petersen, W. (eds.): Frames and Concept Types, Studies in Linguistics and Philosophy, vol. 94. Springer International Publishing (2014), DOI: 10.1007/978-3-319-01541-5
12. Kallmeyer, L., Osswald, R.: Syntax-driven semantic frame composition in lexicalized tree adjoining grammars. Journal of Language Modelling 1(2), 267–330 (2013), DOI: 10.15398/jlm.v1i2.61
13. Kallmeyer, L., Richter, F.: Quantifiers in frame semantics. In: Morrill, G., Muskens, R., Osswald, R., Richter, F. (eds.) Formal Grammar, pp. 69–85. No. 8612 in Lecture Notes in Computer Science, Springer (2014), DOI: 10.1007/978-3-662-44121-3_5
14. Lichte, T., Petitjean, S.: Implementing semantic frames as typed feature structures with XMG. Journal of Language Modelling (to appear)
15. Löbner, S.: Evidence for frames from human language. In: Gamerschlag et al. [11], pp. 23–67, DOI: 10.1007/978-3-319-01541-5_2
16. Muskens, R.: Data semantics and linguistic semantics. In: Aloni, M., Franke, M., Roelofsen, F. (eds.) The dynamic, inquisitive, and visionary life of ϕ, $?\phi$, and $\Diamond\phi$, chap. 24, pp. 175–183. Pumbo.nl (2013), http://www.illc.uva.nl/Festschrift-JMF/papers/23_Muskens.pdf
17. Osswald, R., Van Valin, Jr., R.D.: Framenet, frame structure, and the syntax-semantics interface. In: Gamerschlag et al. [11], chap. 6, pp. 125–156, DOI: 10.1007/978-3-319-01541-5_6
18. Petersen, W.: Representation of concepts as frames. The Baltic International Yearbook of Cognition, Logic and Communication 2, 151–170 (2007), http://user.phil-fak.uni-duesseldorf.de/~petersen/paper/Petersen2007_proof.pdf
19. Schneider, T.: The Complexity of Hybrid Logics over Restricted Classes of Frames. Ph.D. thesis, University of Jena, Germany (2007), http://www.cs.man.ac.uk/%7Eschneidt/publ/sch07_phd.pdf

An Overview on Portuguese Nominalizations

Livy Real[1] and Alexandre Rademaker[2]

[1] IBM Research (Brazil)
[2] IBM Research - FGV/EMAp (Brazil)

Abstract. We discuss nominalizations in Portuguese formed by the suffix *-ura*. We have done a corpus-based description of the behavior of these nominals and proposed a type ontology to categorize them. In order to offer a rich description, we also tested all words formed by *-ura* in co-predication contexts to check if their types could be co-predicated. Although our main goal was to produce a corpus-based description on those nominals, we have found that may be the frequency of use of a given word has a special role on the acceptability of co-predication between different senses of a nominalization.

1 Introduction

If we consider the last decades of formal linguistic studies, the behavior of nominalizations is a very recurring topic, specially because it imposes challenges to lexical semantics theories, knowledge representation systems, and other areas of formalism.

Generally, studies on nominalizations consider only prototypical nominals (as *construction*, *destruction* and *translation*) in the search for generalizations on their behavior or even while deciding what is the best way to represent (or to understand) them. This research intents to look extensively into one specific kind of nominalization: deverbal nouns formed by suffix *-ura* in Brazilian Portuguese (henceforth BP) as *abertura* 'opening', *assinatura* 'signing/signature' and *brancura* 'whiteness'. We hope to reach an enriched description of a relevant fragment of deverbal nominals in BP, considering all possible senses of each noun and possible co-predications between them.

2 Our motivations

We focus on BP because it is not a very mainstream language used in linguistic studies, even though it has similarities with most neo-Latin languages. The choice of the suffix *-ura* was made for two reasons: we already know [5] how this suffix morphologically works and, more than that, the discussion of their behavior in BP probably could be used to understand other suffixes in other neo-Latin languages (as *-ura* in Catalan or *-ure* in French) as they have similar suffixes working on deverbal nominalization processing.

The decision to work on nominalizations formed by a specific suffix was because we want to do a corpus-based overview of one fragment of nominalizations

in order to include action nominals — deverbal nouns which carry eventive readings, cf. [9] — and also non-prototypical nominalizations, that is, nominalizations whose main value is not an eventive or processual meaning.

We hope that our investigation on *-ura* nominals can hint us on the behavior of similar nominalizations in other languages, since *-ura* is a morpheme which has at least eight meanings cataloged by literature on Portuguese ([8, 7, 4]), which include eventive, resultative, locative and collective readings, just as other well-known morphemes like *-ung* in German and *-age* in French. In those lines, we expect that working on these nominal will get new perceptions on what is going on nominalizations.

3 Methodology

To produce an empirical description of this data, we have chosen to work on a corpus-based analysis. All of our descriptions are based on written corpora and the test sentences were checked in a given context with at least three speakers not related to linguistics studies. We believe that it is important to enlarge the discussion on nominalizations since many researches take into account just very well known and simple sentences and constructions.

All the nouns used in the research came from OpenWordNet-PT([3], `http://www.logics.emap.fgv.br/wn/`). We extracted from the list all nominals finished by the graphic form "ura" (442 words) and manually selected the nouns formed by the suffix *-ura* (150 words). It is important to note that, as we extracted synsets from OpenWordnet-PT, some words were counted more than once, as they appear in more than one synset.

Then, to categorize the selected nominals, we have checked dictionaries (Porto's Dictionaries (`http://www.infopedia.pt`), Caldas Aulete Dictionary (`http://www.aulete.com.br`) and Houaiss Dicionary (`http://www.houaiss.uol.com.br`) for all the possible meanings to each noun in our list. We also checked the presence of each noun on Corpus Brasileiro (developed by Tony Berber Sardinha alii) trough AC/DC tool ([2], `http://www.linguateca.pt/ACDC`) which has more than 1 billion words extracted from various textual genders. Google engine was used when Corpus Brasileiro was not enough.

In addition we have relied in recent literature on nominalizations and BP to capture some more insight about the typology those nouns could assume. Almost all the sentences analyzed in this work comes from Corpus Brasileiro and some of them were found on Google search engine in different trustworthy websites.

4 Analysis

Earlier studies have shown that eventive nominalizations in Portuguese can assume at least the following readings (cf. [4]): event, result, physical result, resultative state, abstract result, locative and collective.

Following this categorization, we checked on dictionaries and confirmed on corpora the possible meanings to each noun from our list. From that analysis,

we found that nominals formed by *-ura* can have the following readings: event, result, physical result, locative, collective, means, property, instrument, a given portion, rest, function, duration of a function, science/art, as shown by the examples bellow.

Event Deduziu-se que a mãe lhe deu muita chicotada a cada **travessura**. 'It was deduced that the mother gave him a lot of whiplashes at every **trick** (every time he misbehaved).'

Result A análise do material revelou que, 30 dias após a microenxertia, ocorreu a **soldadura** parcial dos microenxertos. 'The analysis of the material showed that, 30 days after micrografting, occurred the partial **welding** of micrografts.'

Physical Result A **varredura** mostra somente picos, como pode ser visto na Figura 8, onde o espelho de simetria de 0 é mostrado. 'The **scan** shows only peaks, as it can be seen in Figure 8, where the symmetry mirror of 0 is shown.'

Locative Meu certificado está na pasta com meus documentos na **prefeitura**, mas o prefeito não o reconheceu. 'My certificate is the folder with my documents in the **city hall**, but the Mayor did not recognized it.'

Collective Uffizi tem o mais completo testemunho do século XV, um momento decisivo da história da arte, marcado pela passagem da tradição bizantina medieval para a **pintura** do Renascimento. 'Uffizi has the most complete reference of XV Century, a decisive moment of Art History, marked by the passage of Medieval Bizantine tradition to the Renaissance **painting**'

Means A narrativa é um cavalo: um meio de transporte cujo tipo de **andadura**, trote ou galope, depende do percurso a ser executado. 'The narrative is a horse: a means of transportation which type of **gait**, trot or gallop, depends on the route to run.'

Property Possui cerca de 48% de umidade e 24% de **gordura**. 'It has around 48% of umidity and 24% of **fat**.'

Instrument Caricaturizada, a gostosona desfila engravatada, com chapéu, **abotoadura** e tudo mais. 'Caricatured, the hot girl parades with tie, hat, **cufflink** and everything.'

A given portion Assim verificamos que os 587 pés que aquelas dez propriedades dos Calça Pereira comportam podiam render uma mdia de 23,5 **moeduras**, isto , uns 940 alqueires de azeite, que valeriam, ao preço de 60 reais o alqueire, 5640 reais. 'Thus we have verified that the 587 feet of those ten properties from Cala Pereira family include could yield an average of 23.5 **milling portions**, that is, some 940 acres of olive, which would be worth at the price of 60 reais per bushel, 5640 reais.'

Rest O arroz-caril, confeccionado com especiarias e **moedura** de coco, era característico de Goa e estava muito difundido em Moçambique.[3] 'The rice-curry, made with spices and coconut **grinding**, was characteristic of Goa and was widespread in Mozambique.'

[3] http://www.scielo.br/pdf/hcsm/v21n2/0104-5970-hcsm-21-2-0609.pdf

Function Mario renunciou à **magistratura** em novembro. 'Mario resigned to the **magistracy** in November.'

Duration of a function Para a **legislatura** de 1995-1998, os dados provêm do Brasil. 'For the **legislative period** 1995-1998, the data comes from Brazil.'

Science/Art A Itália exprimiu-se, durante certos séculos, pela **arquitetura, escultura, pintura**. 'Italy expressed herself, during some centuries, by the **architecture, sculpture, painting**.'

We also cataloged all the lexicalized values, which are not closely related to the verb base or that cannot be found recurrently in the lexicon. From our 150 nouns list, 33 presented a lexicalized and idiosyncratic meaning. It is worth knowing that two possible types of action nominal formed by *-ura* are not possible (or frequent) in BP: resultative state and abstract result.

Within this categorization, we looked for possible generalizations on the behavior of those nominals and also specific cases that can confirm (or not) the patterns brought by the literature.

After this categorization, we searched for co-predications within the nouns on Corpus Brasiliero and Google. For our surprise, we did not find any co-predication among the nouns that are not used frequently in BP, we only found co-predications among very commonly used nouns (as *assinatura* "signature/signing" and *abertura* "opening").

After that, we produced some sentences combining different types of nouns of frequent and not so frequent use to discover what kind, if any kind, of co-predication would be possible. We have tested all these sentences in context with at least three native speakers of BP with no knowledge of linguistic study theories. Surprisingly again, all the words which are not very commonly used in BP do not accept co-predication in any situation, even when the tested types were 'result' and 'event':

Apesar de ter durado uma hora, a abertura foi proveitosa a todos os alunos.
'Although it lasted one hour, the opening was beneficial to all students.'

*Apesar de ter durado uma hora, a brochura deixou os livros lindos.
'Although it lasted one hour, the brochure left the books beautiful.'

*A mordedura foi rápida, mas deixou uma cicatriz.
'The bite was quick, but left a scar.'

The examples above show co-predication attempts with the words *abertura*, *brochura* and *mordedura* which have a very different number of occurrence in Corpus Brasileiro: *abertura* occurs 70699 times, *brochura* appears 371 times and *mordedura* has only 120 tokens. Therefore, it seems that the frequency of use of a given noun has some influence on its co-predication acceptability, since *abertura* and *mordedura* have the exact same type structure and their verb bases, *abrir* 'open' and *morder* 'bite', have a very similar behavior.

From all this, the generalizations we could get are related to the type structure assumed:

1. A nominal form that has the type 'rest' belongs to the type 'event' (as *lavadura* 'washing' and *varredura* 'scan'), but co-predications between them are impossible;

2. A noun that belongs to the type 'a given portion' (as *moedura* 'milling' and *semeadura* 'sowing') has always the following types: 'event', 'result', 'event.result', but any co-predication with 'a given portion' is blocked;

3. Every noun that belongs to 'duration of a function' also holds the same 'function' type;

4. Nouns that belong to the type 'means' do not belong in any other type;

5. All lexicalized senses can not be co-predicated with any other type.

5 Conclusions and future work

From this small experiment, we can conclude that some nominalizations have a very strict behavior and some others are much more flexible. It seems that another linguistic phenomena could have an influence on this flexibility: the (high or low) frequency that a word is used may be the key to its co-predication acceptability. Besides, our generalizations above are about how particular nominals behave in BP. Perhaps our main contribution with this small analysis is to call the attention to the fact that ordinary nominalizations (highly used words) have the tendency to allow co-predication between their types. Meanwhile uncommon words, with the same type structure and the very same morphological formation, do not.

For this note, we do not present a formalization of our results, but, as argued by [4], it seems that a tool like Montagovian Generative Lexicon is able to deal with this very idiosyncratic behavior of nominalizations formed by *-ura*, as it has a very flexible mechanism to describe (in)felicitous co-predication. For now, the application of Montagovian Generative Lexicon on *-ura* nominalizations and a deeper investigation on the relation between felicitous co-predication and the frequency of use of a given noun remains for future analysis.

References

1. REAL, L. RETORÉ C. *Deverbal Semantics and the Montagovian Generative Lexicon Λ Tyn.* Journal of Logic, Language and Information, v. 1, p. 1, 201, 2014.
2. SANTOS, D. SARMENTO, L. *O projecto AC/DC: acesso a corpora/disponibilização de corpora*, APL, pp. 705-717, Porto, 2002.
3. de Paiva, V. RADEMAKER, A. MELO, G. *OpenWordNet-PT: An Open Brazilian WordNet for Reasoning*, Proceedings of the 24th International Conference on Computational Linguistics, 2012.
4. REAL, L. *Nominalizações.* Ph.D thesis, Universidade Federal do Paraná, Curitiba, Brazil, 2014.
5. REAL, L. *Morfologia Categorial.* Undergraduation final work, Universidade Federal do Paraná, Curitiba, Brazil, 2006.

6. BASSAC, C. MERY, B. RETORÉ, C. *A Montagovian generative lexicon* in CSLI, Formal Grammar 2007.
7. ROCHA, L. C. *A nominalização no português do Brasil.* Revista de Estudos da Linguagem, 8 (17), 1999.
8. SANDMANN, A. J. *Formação de palavras no português brasileiro contemporâneo.* Curitiba: Scentia et Labor: Ícone, 1988.
9. MELLONI, C. *Polysemy in word formation: the case of deverbal nominals.* University of Verona: Dissertation, 2007

Formalising type-logical grammars in Agda

Pepijn Kokke

Utrecht University

Abstract. In recent years, the interest in using proof assistants to formalise and reason about mathematics and programming languages has grown. Type-logical grammars, being closely related to type theories and systems used in functional programming, are a perfect candidate to next apply this curiosity to. The advantages of using proof assistants is that they allow one to write formally verified proofs about one's type-logical systems, and that any theory, once implemented, can immediately be computed with. The downside is that in many cases the formal proofs are written as an afterthought, are incomplete, or use obtuse syntax. This makes it that the verified proofs are often much more difficult to read than the pen-and-paper proofs, and almost never directly published. In this paper, we will try to remedy that by example.

Concretely, we use Agda to model the Lambek-Grishin calculus, a grammar logic with a rich vocabulary of type-forming operations. We then present a verified procedure for cut elimination in this system. Then we briefly outline a continuation-passing style translation from proofs in the Lambek-Grishin calculus to programs in Agda. And finally, we will put our system to use in the analysis of a simple example sentence.

1 Introduction

Why would we want to formalise type-logical grammars using proof assistants? One good reason is that it allows us to write formally verified proofs about the theoretical properties of our type-logical grammars. But not only that—it allows us to directly run our proofs as programs. For instance, we can directly run the procedure for cut elimination in this paper to investigate what kind of derivations are created by it *and* be confident in its correctness.

Why, then, would we want to use Agda instead of a more established proof assistant such as, for instance, Coq? There are several good reasons, but we believe that the syntactic freedom offered by Agda is the most important. It is this freedom that allows us to write machine-checkable proofs, formatted in a way which is very close to the way one would otherwise typeset proofs, and which are highly readable compared to other machine-checked proofs. This means that we can be confident that the proofs *as they are published* are correct, and that they are necessarily complete—for though we can hide some of the less interesting definitions from the final paper, we cannot omit them from the source.

Additionally, because there is a one-to-one correspondence between the published proofs and the code, it becomes very easy for the reader to start up a proof environment and inspect the proofs interactively in order to further their understanding.

Our test case in this paper is the Lambek-Grishin calculus (LG, **?**). LG is an example of an extended Lambek calculus. In addition to the product ($\otimes$) and the residual slashes ($\backslash$, $/$), LG has a dual family with $\oplus$ and difference operations ($\oslash$, $\obslash$) together with distributivity principles for the interaction between the two families. See **?** for discussion of how LG overcomes syntactic and semantic limitations of the original Lambek calculus.

We will formalise the residuation-monotonicity axiomatisation for the Lambek-Grishin calculus (**?**) in Agda, present a verified procedure for cut elimination in this system, and briefly outline a continuation-passing style (CPS) translation into Agda. There are several reasons why we have chosen to formalise this particular system.

- It allows cut as an admissible rule, i.e. a function on proofs, instead of defining a separate cut-free system and a cut-elimination procedure;
- it has efficiently decidable proof search, largely due to the absence of the cut rule;
- it has some interesting symmetries, as explored in **??**. Because of this, most proofs of properties of LG are not much more complicated than their associated proofs for the non-associative Lambek calculus;
- it has a continuation-passing style interpretation, which has shown to be useful in both derivational and lexical semantics (**???**);
- lastly, an implementation of the non-associative Lambek calculus can easily and mechanically be extracted from our implementation of LG.

Since this paper is by no means a complete introduction to Agda or to dependently-typed programming, we advise the interested reader to refer to **?** for a detailed discussion of Agda.

It should be mentioned that (although we omit some of the more tedious parts) this paper is written in literate Agda, and the code has been made available on GitHub.[1]

[1] See https://gist.github.com/pepijnkokke/cc12b92a8a60696b712c#file-main-agda.

2 Formulas, Judgements, Base System

If we want to model our type-logical grammars in Agda, a natural starting point would be our atomic formulas—such as n, np, s, etc. These could easily be represented as an enumerated data type. However, in order to avoid committing to a certain set of atomic formulas and side-step the debate on which formulas *should* be atomic, we will simply assume there is a some data type representing our atomic formulas. This will be reflected in our module header as follows:

```
module logic (Atom : Set) where
```

Our formulas can easily be described as a data type, injecting our atomic formulas by means of the constructor el, and adding the familiar connectives from the Lambek-Grishin calculus as binary constructors. Note that, in Agda, we can use underscores in definitions to denote argument positions. This means that _⊗_ below defines an infix, binary connective:

```
data Type : Set where
  el                  : Atom → Type
  _⊗_ _↖_ _↗_ : Type → Type → Type
  _⊕_ _⊘_ _⊖_ : Type → Type → Type
```

In the same manner, we can define a data type to represent judgements:

```
data Judgement : Set where
  _⊢_ : Type → Type → Judgement
```

Using the above definitions, we can now write judgements such as A ⊗ A ↖ B ⊢ B as Agda values. Next we will define a data type to represent our logical system. This is where we can use the dependent type system to our advantage. The constructors for our data type will represent the axiomatic inference rules of the system, and their *types* will be constrained by judgements. Below you can see the entire system LG as an Agda data type[2]:

```
data LG_ : Judgement → Set where
  ax    : LG el A ⊢ el A
    -- residuation and monotonicity for (↗ , ⊗ , ↖)
  r↖⊗ : LG B ⊢ A ↖ C → LG A ⊗ B ⊢ C
  r⊗↖ : LG A ⊗ B ⊢ C → LG B ⊢ A ↖ C
  r↗⊗ : LG A ⊢ C ↗ B → LG A ⊗ B ⊢ C
  r⊗↗ : LG A ⊗ B ⊢ C → LG A ⊢ C ↗ B
  m⊗  : LG A ⊢ B → LG C ⊢ D → LG A ⊗ C ⊢ B ⊗ D
  m↖  : LG A ⊢ B → LG C ⊢ D → LG B ↖ C ⊢ A ↖ D
  m↗  : LG A ⊢ B → LG C ⊢ D → LG A ↗ D ⊢ B ↗ C
    -- residuation and monotonicity for (⊘ , ⊕ , ⊖)
  r⊖⊕ : LG B ⊖ C ⊢ A → LG C ⊢ B ⊕ A
  r⊕⊖ : LG C ⊢ B ⊕ A → LG B ⊖ C ⊢ A
  r⊕⊘ : LG C ⊢ B ⊕ A → LG C ⊘ A ⊢ B
  r⊘⊕ : LG C ⊘ A ⊢ B → LG C ⊢ B ⊕ A
  m⊕  : LG A ⊢ B → LG C ⊢ D → LG A ⊕ C ⊢ B ⊕ D
  m⊖  : LG C ⊢ D → LG A ⊢ B → LG D ⊖ A ⊢ C ⊖ B
  m⊘  : LG A ⊢ B → LG C ⊢ D → LG A ⊘ D ⊢ B ⊘ C
    -- grishin distributives
  d⊖↗ : LG A ⊗ B ⊢ C ⊕ D → LG C ⊖ A ⊢ D ↗ B
  d⊖↖ : LG A ⊗ B ⊢ C ⊕ D → LG C ⊖ B ⊢ A ↖ D
  d⊘↖ : LG A ⊗ B ⊢ C ⊕ D → LG B ⊘ D ⊢ A ↖ C
  d⊘↗ : LG A ⊗ B ⊢ C ⊕ D → LG A ⊘ D ⊢ C ↗ B
```

Note that Agda allows arbitrary unicode characters in identifiers, so r⊗↖ is a valid Agda identifier.

Using this data type, we can already do quite a lot. For instance, we can show that while the inference rule ax above is restricted to atomic formulas[3], the unrestricted version is admissible, by induction on the formula. Note that the construction {A = ...} below is used to pattern match on the implicit variable A:

[2] For the typeset version of this paper we omit the quantifiers for all implicit, universally quantified arguments.

[3] Whereas the rule ax may appear to be unrestricted, it only allows the derivation of the identity proof for any formula el A. That is, any *atomic formula* A delimited by the constructor el.

```
ax′ : LG A ⊢ A
ax′ {A = el   _} = ax
ax′ {A = _ ⊗ _} = m⊗ ax′ ax′
ax′ {A = _ ↗ _} = m↗ ax′ ax′
ax′ {A = _ ↖ _} = m↖ ax′ ax′
ax′ {A = _ ⊕ _} = m⊕ ax′ ax′
ax′ {A = _ ⊘ _} = m⊘ ax′ ax′
ax′ {A = _ ⊖ _} = m⊖ ax′ ax′
```

Alternatively, we could derive the various applications and co-applications that hold in the Lambek-Grishin calculus:

```
appl-↖′ : LG A ⊗ (A ↖ B) ⊢ B
appl-↖′ = r↖⊗ (m↖ ax′ ax′)
appl-↗′ : LG (B ↗ A) ⊗ A ⊢ B
appl-↗′ = r↗⊗ (m↗ ax′ ax′)
appl-⊖′ : LG B ⊢ A ⊕ (A ⊖ B)
appl-⊖′ = r⊖⊕ (m⊖ ax′ ax′)
appl-⊘′ : LG B ⊢ (B ⊘ A) ⊕ A
appl-⊘′ = r⊘⊕ (m⊘ ax′ ax′)
```

However, the most compelling reason to use the axiomatisation we have chosen, using residuation and monotonicity rules, is that cut becomes an admissible rule.

3 Admissible Cut

We would like to show that cut′ of type $LG\ \mathsf{A} \vdash \mathsf{B} \to LG\ \mathsf{B} \vdash \mathsf{C} \to LG\ \mathsf{A} \vdash \mathsf{C}$ is an admissible rule. The method of **?**, for the basic non-associative Lambek calculus, can be readily generalized to the case of LG:

(i) every connective is introduced *symmetrically* by a monotonicity rule or as an axiom;
(ii) every connective has one side (antecedent or succedent) where, if it occurs there at the top level, it cannot be taken apart or changed by any inference rule;
(iii) as a consequence of (ii), every formula has one side where, if it occurs there at the top level, it is immutable, i.e. there is no rule which can eliminate it;
(iv) due to (i) and (iii), when we find such an immutable formula, we can be sure that, stepping through the derivation, after some number of steps we will find the monotonicity rule which introduced that formula;
(v) due to the type of cut′, when we match on the cut formula B we will always have an immutable variant of that formula in either the first or the second argument of cut′;
(vi) finally, for each main connective there exists a rewrite rule which makes use of the facts in (iv) and (v) to rewrite an application of cut′: to two applications of cut′ on the arguments of the monotonicity rule which introduced the connective, chained together by applications of residuation (for binary connectives) or simply to a derivation (for atomic formulas). As an example, the rewrite rule for _⊗_ can be found in figure 1.

$$\dfrac{\dfrac{\dfrac{E \vdash B \qquad F \vdash C}{E \otimes F \vdash B \otimes C}}{\vdots}}{\dfrac{A \vdash B \otimes C \qquad B \otimes C \vdash D}{A \vdash D}} \quad \rightsquigarrow \quad \dfrac{E \vdash B \qquad \dfrac{\dfrac{F \vdash C \qquad \dfrac{B \otimes C \vdash D}{C \vdash B \nwarrow D}}{F \vdash B \nwarrow D}}{B \otimes F \vdash D}}{\dfrac{\dfrac{B \vdash D \nearrow F}{E \vdash D \nearrow F}}{\dfrac{E \otimes F \vdash D}{\vdots}}}$$

Fig. 1. Rewrite rule for cut on formula B ⊗ C.

We can model the view on the left-hand side of the rewrite rule in figure 1 as a data type. In order to construct this view for some suitable derivation f, we need two derivations, h_1 and h_2 and a derivation f′, which represents the arbitrary derivation steps taking (m⊗ h_1 h_2) back to f. Lastly, we include a proof pr of the fact that the reconstructed derivation f′ (m⊗ h_1 h_2) is identical to f:

```
data Origin (f : LG A ⊢ B ⊗ C) : Set where
  origin : (h₁ : LG E ⊢ B)
           (h₂ : LG F ⊢ C)
           (f′ : ∀ {G} → LG E ⊗ F ⊢ G → LG A ⊢ G)
           (pr : f ≡ f′ (m⊗ h₁ h₂))
             → Origin f
```

In the above snippet, we have chosen to leave the quantifier $\forall\ \{\mathsf{G}\}$ explicit to stress that f' should work for *any* formula G, not only for $\mathsf{B} \otimes \mathsf{C}$.

All that remains now is to show that for any f of type $LG\ \mathsf{A} \vdash \mathsf{B} \otimes \mathsf{C}$, we can construct such a view. We will attempt to do this by induction on the given derivation. Note that { }0 is the Agda syntax for a proof obligation. For clarity, I have added the types of the various subproofs f in comments:

```
find : (f : LG A ⊢ B ⊗ C) → Origin f
find (m⊗ f g) = origin f g id refl
find (r↖⊗ f)  = { }0  -- f : LG A₂ ⊢ A₁ ↖ B ⊗ C
find (r↗⊗ f)  = { }1  -- f : LG A₁ ⊢ B ⊗ C ↗ A₂
find (r⊕⊘ f)  = { }2  -- f : LG A₂ ⊢ A₁ ⊕ B ⊗ C
find (r⊕⊘ f)  = { }3  -- f : LG A₁ ⊢ B ⊗ C ⊕ A₂
```

Alas! While in the first case, where f is of the form m⊗ f g, we have found our monotonicity rule, the remaining cases are less kind. It seems that we have neglected to account for derivations where our cut formula is temporarily nested within another formula.

We will need some new vocabulary to describe what is going on in the above example. We would like to describe contexts which a) can be taken apart using residuation, and b) when fully taken apart, will leave the nested formula on the correct side of the turnstile. A natural fit for this is using polarity:

```
data Polarity : Set where + − : Polarity
```

Below we define well-polarised formula and judgement contexts with exactly one hole. We use a ◃ or ▹ to denote in which argument the hole is:

```
data Context (p : Polarity) : Polarity → Set where
  []       : Context p p
  _⊗▹_     : Type → Context p +  → Context p +
  _↖▹_     : Type → Context p −  → Context p −
  _↗▹_     : Type → Context p +  → Context p −
  _◃⊗_     : Context p +  → Type → Context p +
  _◃↖_     : Context p +  → Type → Context p −
  _◃↗_     : Context p −  → Type → Context p −
  _⊕▹_     : Type → Context p −  → Context p −
  _⊘▹_     : Type → Context p −  → Context p +
  _⊘▹_     : Type → Context p +  → Context p +
  _◃⊕_     : Context p −  → Type → Context p −
  _◃⊘_     : Context p +  → Type → Context p +
  _◃⊘_     : Context p −  → Type → Context p +
data Context^J (p : Polarity) : Set where
  _◃⊢_     : Context p +  → Type → Context^J p
  _⊢▹_     : Type → Context p −  → Context^J p
```

We also define two operators which, given a context and a formula, will fill the hole in the given context with the given formula. The definition for _ [_] is entirely predictable and repetitive, and has been mostly omitted[4]:

```
_ [_] : Context p₁ p₂ → Type → Type
[]          [A] = A
(B ⊗▹ C) [A] = B ⊗ (C [A])
...
```

[4] For the remainder of this paper, any partial omission of a function will be denoted with an ellipsis at the end of the code block.

```
_ [_] ^J : Context^J p → Type → Judgement
(A ◃ ⊢ B) [C] ^J = A [C] ⊢ B
(A ⊢ ▹ B) [C] ^J = A ⊢ B [C]
```

The crucial point about these well-polarised judgement contexts is that, once the entire context is peeled away, the formula will be at the top level on the side corresponding to the polarity argument—with + and − corresponding to the antecedent and the succedent, respectively. Therefore, in order to generalise our previous definition of Origin, we want the occurrence of B ⊗ C to be nested in a *negative* context:

```
data Origin′ (J : Context^J −)
     (f  : LG J [B ⊗ C] ^J)
         : Set where
  origin : (h₁ : LG E ⊢ B)
     (h₂ : LG F ⊢ C)
     (f′ : LG E ⊗ F ⊢ G → LG J [G] ^J)
     (pr : f ≡ f′ (m⊗ h₁ h₂))
         → Origin′ J f
```

Using this more general definition Origin′, we can define a more general function find′—and this time, our proof by induction works!

Note that in Agda, the **with** construct is used to pattern match on the result of an expression:

```
find′ : (J : Context^J −) (f : LG J [B ⊗ C] ^J) → Origin′ J f
find′ (._ ⊢ ▹ [])          (m⊗ f g) = origin f g id refl
find′ (._ ⊢ ▹ (A ◃ ⁄ _)) (r⊗⁄ f)  with find′ (_ ⊢ ▹ A) f
... | origin h₁ h₂ f′ pr rewrite pr   = origin h₁ h₂ (r⊗⁄ ∘ f′) refl
find′ (._ ⊢ ▹ (_ ⧵ ▹ B)) (r⊗⧵ f)  with find′ (_ ⊢ ▹ B) f
... | origin h₁ h₂ f′ pr rewrite pr   = origin h₁ h₂ (r⊗⧵ ∘ f′) refl
...
```

However, there are many cases—53 in total. The reason for this is that the possible derivation steps depend on the main connective; therefore we first have to explore every possible main connective, and then every possible rule which would produce that main connective. Because of this, the definitions of the various find′ functions are very long and tedious, and have mostly been omitted.[5]

From the more general Origin′ and find′ we can very easily recover our original definitions Origin and find by setting the context to be empty. In the case of the cut formula B ⊗ C, we set the context to (_ ⊢ ▹ []) to ensure that the formula ends up at the top level in the succedent:

```
Origin : (f : LG A ⊢ B ⊗ C) → Set
Origin f = Origin′ (_ ⊢ ▹ []) f
find   : (f : LG A ⊢ B ⊗ C) → Origin f
find   f = find′   (_ ⊢ ▹ []) f
```

And with that, we can finally put the rewrite rules from **?** to use. We can define cut′ by pattern matching on the cut formula B; applying the appropriate find′ function to find′ the monotonicity rule introducing the formula; and apply the appropriate rewrite rule to create a derivation containing two cuts on structurally smaller formulas:

```
cut′ : (f : LG A ⊢ B) (g : LG B ⊢ C) → LG A ⊢ C
cut′ {B = el _}   f g with el.find g
... | (el.origin          g′ _) = g′ f
cut′ {B = _ ⊗ _} f g with ⊗.find f
... | (⊗.origin     h₁ h₂ f′ _) = f′ (r⁄⊗ (cut′ h₁ (r⊗⁄ (r⧵⊗ (cut′ h₂ (r⊗⧵ g))))))
cut′ {B = _ ⁄ _} f g with ⁄.find g
... | (⁄.origin     h₁ h₂ g′ _) = g′ (r⊗⁄ (r⧵⊗ (cut′ h₂ (r⊗⧵ (cut′ (r⁄⊗ f) h₁)))))
cut′ {B = _ ⧵ _} f g with ⧵.find g
... | (⧵.origin     h₁ h₂ g′ _) = g′ (r⊗⧵ (r⁄⊗ (cut′ h₁ (r⊗⁄ (cut′ (r⧵⊗ f) h₂)))))
cut′ {B = _ ⊕ _} f g with ⊕.find g
```

[5] The burden on the programmer or logician can be reduced by clever use of the symmetries $\cdot^{\bowtie}$ and $\cdot^{\infty}$ as done in **?**. One would have to implement only *three* of the find′ functions (e.g. for el, ⊗ and ⧵); the remaining four can then be derived using the symmetries.

```
... | (⊕.origin      h₁ h₂ g′ _) = g′ (r⊘⊕ (cut′ (r⊕⊘ (r⊘⊕ (cut′ (r⊕⊘ f) h₂))) h₁))
cut′ {B = _ ⊘ _} f g with ⊘.find f
... | (⊘.origin      h₁ h₂ f′ _) = f′ (r⊕⊘ (r⊘⊕ (cut′ (r⊕⊘ (cut′ h₁ (r⊘⊕ g))) h₂)))
cut′ {B = _ ⊘ _} f g with ⊘.find f
... | (⊘.origin      h₁ h₂ f′ _) = f′ (r⊕⊘ (r⊘⊕ (cut′ (r⊕⊘ (cut′ h₂ (r⊘⊕ g))) h₁)))
```

4 CPS Translation

For this paper, we have opted to implement the call-by-value CPS translation as described in **?**. This translation consists of three elements:

- a function $\lceil \_ \rceil$, which translates formulas in LG to formulas in the target system—while we have chosen to translate to Agda, the original translation targeted multiplicative intuitionistic linear logic;
- a pair of mutually recursive functions $\lceil \_ \rceil^L$ and $\lceil \_ \rceil^R$, which translate terms in LG to terms in the target system.

In order to write these functions, we will need two additional pieces of information: a function $\lceil \_ \rceil^A$, which translates the atomic formulas to Agda types; and a return type R, which we will use to define a "negation" as $\neg\, A = A \to R$. We will therefore implement the CPS translation in a sub-module, which abstracts over these terms:

```
module translation (⌈_⌉ᴬ : Atom → Set) (R : Set) where
```

When using this module, we will generally identify the return type R with the type Bool for booleans. However, abstracting over it will ensure that we do not accidentally use this knowledge during the translation.

The type-level translation itself maps formulas in LG to types in Agda, as follows:

```
⌈_⌉ : Type → Set
⌈ el   A ⌉ =       ⌈ A ⌉ᴬ
⌈ A ⊗ B ⌉ =    (   ⌈ A ⌉ ×   ⌈ B ⌉)
⌈ A ↖ B ⌉ = ¬ (   ⌈ A ⌉ × ¬ ⌈ B ⌉)
⌈ B ↗ A ⌉ = ¬ (¬ ⌈ B ⌉ ×   ⌈ A ⌉)
⌈ B ⊕ A ⌉ = ¬ (¬ ⌈ B ⌉ × ¬ ⌈ A ⌉)
⌈ B ⊘ A ⌉ =    (   ⌈ B ⌉ × ¬ ⌈ A ⌉)
⌈ A ⊘ B ⌉ =    (¬ ⌈ A ⌉ ×   ⌈ B ⌉)
```

The translations on terms map terms in LG to the Agda function space. Each LG term is associated with *two* functions, depending on whether the focus is on A or B as the active formula:

```
mutual
  ⌈_⌉ᴸ : LG A ⊢ B → ¬ ⌈ B ⌉ → ¬ ⌈ A ⌉
  ⌈_⌉ᴿ : LG A ⊢ B →   ⌈ A ⌉ → ¬ ¬ ⌈ B ⌉
...
```

The CPS translations of the terms are rather verbose, and trivial to deduce, when guided by the translation on types. Therefore, in the interest of space they have been omitted from the paper.[6]

5 Example

In this final section, we will present the analysis of an example sentence, using the type-logical grammar implemented above. The example we will analyse is:

> "Someone loves everyone."

This sentence is well known to be ambiguous, owing to the presence of the two quantifiers. There are two readings:

a. There is some person who loves every person.
b. For each person, there is some person who loves them.

[6] They are, however, present in the source and therefore available on GitHub.

We will demonstrate that the system, as implemented in this paper, accurately captures these readings.

Before we can do that, however, there is a small amount of boiler plate that we have to deal with: we still need to choose a representation for our atomic types, and show how these translate into Agda. In what follows, we will assume we have access to a type for entities, suitable definitions for the universal and existential quantifiers, and meanings for 'loves' and 'person':

```
postulate
  Entity : Set
  ∀      : (Entity → Bool) → Bool
  ∃      : (Entity → Bool) → Bool
  LOVES  : Entity → Entity → Bool
  PERSON : Entity → Bool
```

We will instantiate the type for atomic formulas to Atom, as defined below:

```
data Atom : Set where N NP S : Atom
```

Last, we need to define a function which maps the values of Atom to Agda types. We would like to map the atomic formulas as follows:

```
⌈_⌉ᴬ : Atom → Set
⌈ N ⌉ᴬ = Entity → Bool
⌈ NP ⌉ᴬ = Entity
⌈ S ⌉ᴬ = Bool
```

Now that we have Atom and $\lceil \_ \rceil^A$, we can open up the modules defined as above, instantiating the return type R with the type of booleans.

```
open logic             Atom
open logic.translation Atom ⌈_⌉ᴬ Bool
```

With everything that we implemented in scope, we can now define a small lexicon for our example sentence.

In what follows, we will use the aliases n, np and s for el N, el NP and el S, respectively:

```
someone : ⌈ (np ↗ n) ⊗ n ⌉
someone = ((λ {(g , f) → ∃ (λ x → f x ∧ g x)}) , PERSON)
loves : ⌈ (np ↖ s) ↗ np ⌉
loves = λ {(k , y) → k (λ {(x , k) → k (LOVES x y)})}
everyone : ⌈ (np ↗ n) ⊗ n ⌉
everyone = ((λ {(g , f) → ∀ (λ x → f x ⊃ g x)}) , PERSON)
```

Given the types we used for our lexical entries, the judgement which asserts the grammaticality of our sentence becomes:

```
((np ↗ n) ⊗ n) ⊗ (((np ↖ s) ↗ np) ⊗ ((np ↗ n) ⊗ n)) ⊢ s
```

There are seven proofs of this judgement. Below we have included the first *two* proofs:[7]:

```
SENT₀ = r↖⊗ (r↗⊗ (m↗ (m↖ (r↗⊗ ax′) ax) (r↗⊗ ax′)))
SENT₁ = r↗⊗ (r↗⊗ (m↗ (r⊗↗ (r↖⊗ (r↗⊗ (m↗ ax′ (r↗⊗ ax′))))) ax))
...
```

We can now apply our CPS translation to compute the denotations of our sentence, passing in the denotations of the words as a tuple, and passing in the identity function as the last argument in order to obtain the result:

```
sent₀ : ⌈ SENT₀ ⌉ᴿ (someone , loves , everyone) id ↦ ∀ (λ y → PERSON y ⊃ ∃ (λ x → PERSON x ∧ LOVES x y))
sent₁ : ⌈ SENT₁ ⌉ᴿ (someone , loves , everyone) id ↦ ∃ (λ x → PERSON x ∧ ∀ (λ y → PERSON y ⊃ LOVES x y))
...
```

Voila! Our system produces exactly the expected readings.

[7] We have chosen not to include the other five proofs as, under the CPS translation, they have the same interpretations as either the first or the second proof. For the interested reader, however, the proofs are present in the source, and therefore available on GitHub.

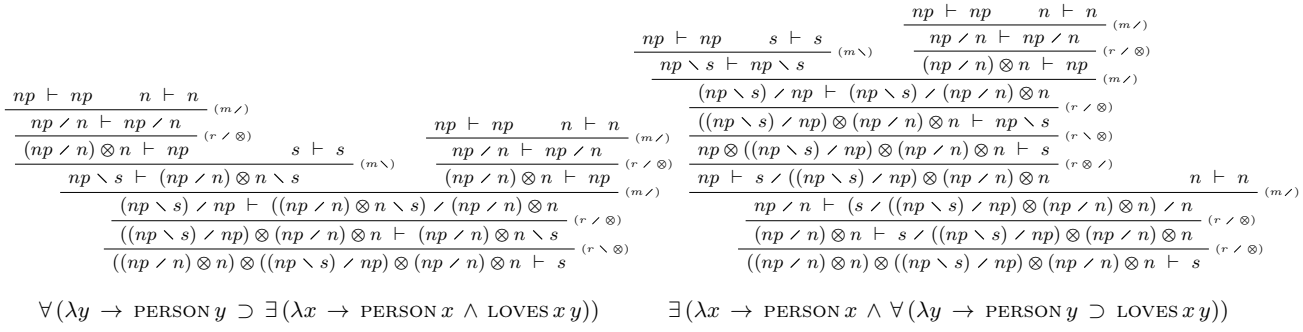


Fig. 2. "Someone loves everyone."

6 Conclusion

We have presented the reader with a simple formalisation of the Lambek-Grishin calculus, using the proof assistant Agda. We have shown how to formalise the proof of the admissibility of cut from **?** in Agda, and have extended this proof to cover all of LG. While we have not covered any of the usual unary operators, the formalism presented here generalises straightforwardly to accommodate connectives of any arity—and this extension, together with many other extensions, are indeed implemented in the full version of our code.

We have then presented the reader with a call-by-value CPS translation into the host language Agda, and used this translation to demonstrate the analysis of an example sentence.

Most importantly, we hope we presented the reader with a clean and readable formalisation of the Lambek-Grishin calculus.

7 Related & Future Work

Previous work on the formalisation of Lambek calculi was done in Coq by **?**, who, amongst other things, implemented sequent calculus and natural deduction systems for multi-modal categorial grammars.

The work presented in this paper is part of a larger undertaking to formalise type-logical grammars in Agda. At the moment, we have formalised not only the algebraic Lambek-Grishin calculus—which was presented in this paper—but also structural and polarised varieties of this calculus. From these implementations, we are able to extract implementations of their respective non-associative Lambek calculi.

In addition, we have implemented various other multi-modal systems, such as NL_{CL} (**?**).

We aim to extend this work by further formalising the known work w.r.t. these calculi, and creating tools to accommodate the writing of formal linguistics papers in literate style.

References

Anoun, H. (2007). Une bibliothèque coq pour le traitement des langues naturelles. *Technique et Science Informatiques*, 26(9):1111–1136.

Anoun, H., Castéran, P., and Moot, R. (2004). Proof automation for type-logical grammars. Rapport de recherche , European Summer School in Logic, Language and Information - 2004.

Asher, N. (2011). *Lexical Meaning in Context: A Web of Words*. Cambridge University Press.

Barker, C. and Shan, C. (2015). *Continuations and Natural Language (Oxford Studies in Theoretical Linguistics)*. Oxford University Press.

Moortgat, M. (1996). In situ binding: a modal analysis. In Dekker, P. and Stokhof, M., editors, *Proceedings of the Tenth Amsterdam Colloquium*, Amsterdam. Institute for Logic, Language and Computation, Amsterdam.

Moortgat, M. (2007). Symmetries in natural language syntax and semantics: The lambek-grishin calculus. In *Logic, Language, Information and Computation*, pages 264–284. Springer Berlin Heidelberg.

Moortgat, M. (2009). Symmetric categorial grammar. *Journal of Philosophical Logic*, 38(6):681–710.

Moortgat, M. and Oehrle, R. T. (1999). Proof nets for the grammatical base logic. In Abrusci, M. and Casadio, C., editors, *Dynamic perspectives in logic and linguistics. Proceedings of the Fourth Roma Workshop*, pages 131–143, Roma. Bulzoni.

Moot, R. and Retoré, C. (2012). *The Logic of Categorial Grammars*. Springer Berlin Heidelberg.

Norell, U. (2009). Dependently typed programming in agda. In *Proceedings of the 4th International Workshop on Types in Language Design and Implementation*, TLDI '09, pages 1–2, New York, NY, USA. ACM.

A DEVELOPED ANALYSIS of TYPE COERCION based on TYPE THEORY and CONVENTIONALITY

Seohyun Im[1] and Chunngmin Lee[2]

[1] Seoul National University seohyunim71@gmail.com
[2] Seoul National University clee@snu.ac.kr

Abstract. This paper aims to propose a developed analysis of the type coercion phenomenon such as *begin the book* by introducing Type Theory and Conventional Non-linguistic Context, making a distinction between linguistic and non-linguistic context. We argue that linguistic and non-linguistic context as well as the lexical meaning of the words are deeply involved in the interpretation of the type-coerced construction. In the lexical semantic level, the type-coerced construction is ambiguous. Although its linguistic context can decrease the number of possible interpretations of the construction, it is still ambiguous until its non-linguistic context disambiguates the meaning of the construction. More importantly, we propose that the lexical meaning of a word is a conventionalized meaning under the assumption of a conventional non-linguistic context linked to the word. The context holds in the compositional process. Therefore, a type-coerced construction has a preferred interpretation derived from its conventional non-linguistic context, if no specific non-linguistic context (situation of utterance) is provided and its linguistic context is neutral. For instance, the preferred interpretation of *begin the book* is *begin reading the book*, because the conventional non-linguistic context of *book* is the situation of reading the book. However, the preference is only a probability and the construction is still ambiguous.

1 Introduction

In this paper, we aim to give a developed analysis of type coercion phenomena such as *begin the book* explained in the early Generative Lexicon theory (GL: [PJ1995], [PJms]). For that purpose, we adopt the type theory based on Type Compositional Logic (TCL: [AN2011]). In addition, we introduce the "conventional non-linguistic context", distinguishing between linguistic and non-linguistic context.

One of the strong points of the GL is to explain well the polymorphic behavior of argument selection by predicates, as shown in (1).

(1) a. John **began** *reading the book*.
b. John **began** *to read the book*.
c. John **began** *the book*.

The GL provides the methodology - qualia structure and type coercion - to explain the polymophism of the *begin* construction and recover the missing predicate in a construction such as (1c). However, the type coercion mechanism of the GL brings up the following questions:

- What conditions allow the type coercion of *begin*?
- What is the mechanism of recovering the missing predicate in the type coercion construction with *begin*?
- What are the effects of context on type coercion and its interpretation?
- What is the relation between type coercion and (lexical) semantics and pragmatics?

We go through the following steps to answer the above questions.

1. We propose the semantic type of *begin* and its selection restriction. (section 2)
2. We propose the semantic type of *book* and its distributional constraint. (section 3)
3. We consider the condition which allows the type coercion.(section 4)
4. We discuss the interpretation of the type-coerced construction and the effects of context. (section 5, 6)
5. We conclude our argument . (section 7)

In the next section, we explore the semantic type of the aspectual verb *begin* in English.

2 The Semantic Type of *BEGIN*

According to the GL, the verb *begin* requires an event-type object argument, because it is an aspectual-type verb. However, this type assignment is not specific enough to explain the type coercion phenomenon of *begin*. In this paper, we divide the aspectual verbs into more specific subtypes, following [IS2013]. The verb *begin* is classified as a begin-type, distinguished from other aspectual verbs such as *finish, continue*, and *stop*. The begin-type verbs select only process or accomplishment-type expressions as their object arguments[3]. Consider the following examples:

(2) a. Kern began **building a house**. (accomplishment)
b. He began **working**. (process)

In (2), *building a house* (2a) is accomplishment-type and and *working* (2b) is process-type (2b). On the contrary, *begin* takes neither an achievement nor a state-type gerundive construction. The sentence in (3a) is ungrammatical because *buying the modern painting* represents an achievement-type eventuality.

[3] We do not discuss here the other subtypes in the aspectual-type verb class.

(3) a. *He had begun *buying* the modern **painting**.
b. He had begun *buying* the modern **paintings**.

In (3b), *begin* allows the gerundive construction in which *buying* takes a plural object, since the gerundive is process type, not achievement type[4]. The argument selection restriction of the begin-type verbs is explained well by its event structure ([IS2013]).

(4) Event Structure of the begin-type verbs
se1: state = not_in_process(e2)
se2: process = beginning(x, e2)
se3: process = in_process(e2)

For an event to be in process after its beginning, it should at least belong to process or accomplishment.

To sum up, we propose the semantic type of *begin* and the argument structure - argument selection restriction - of the begin-type verb class as follows:

(5) *begin*
a. **semantic type** = begin
b. **argument structure** (selection restriction) of begin-type verbs
arg1 = agent: top
arg2 = event: {process, accomplishment}

The argument structure above represents the type selection restriction on the arguments of *begin*. It implies that *begin* cannot take the achievement or state type ones out of the verbs governing the NP *the book*. In the next section, we explore the semantic type of *book* and its governing predicates.

3 The Semantic Type of the Noun *BOOK*

The semantic type of the noun *book* needs to be more specific than phys_object·info, which was proposed in the GL, to cover its distributional behavior, although it is right to consider it as a complex type. In this paper, we propose a tentative semantic type of *book* below.

- The Semantic Type of *book* (tentative proposal)
[text_info]· [info_container_made_by_binding_papers]· [goods]

The dot type of *book* has many entailments related with its linguistic context - expressions co-occuring with the noun *book*. The semantic type text_info shows the entailments below [5]:

[4] This kind of type shifting from achievement to process is a well-known phenomenon.
[5] In this work, we only consider the verbs which take *the book* as their direct object. We will extend the corpus so that we can define the complete semantic type of *book* in the furture work.

- text_info
  $\rightarrow_{entail}$
  [*write*, *read*, *translate*, *publish*, *digitize*, *evaluate*, *underestimate*, *criticize*,*etc.*]

The text_info subtype means that a book is text-type information written in a language and thus is readable and translatable[6]. Moreover, it can be published, digitized, or evaluated because it is a kind of information. The text_info is a subtype of info and thus inherits all the properties of it. The subtype triggers entailments related with the governing predicates of the noun *the book* listed above.

The second subtype info_container_made_by_binding_papers inherits from artifact (artifact $\subset$ physical_object). We show entailments triggered by this type as follows.

- info_container_made_by_binding_papers
  $\rightarrow_{entail}$
  [*take*, *carry*, *put*, *place*, *position*, *pack*, *tear*, *burn*, *weight*, *borrow*, *own*, *have*, *lend*,*etc.*]

The list of verbs includes motion verbs and possession or change_of_possession verbs. Since it is made by binding papers, entailments such as tearing are allowed.

Finally, the subtype goods entails all events related to buying or selling. We show the entailments below:

- goods
  $\rightarrow_{entail}$
  [*buy*, *sell*, *promote*, *market*, *advertise*,*etc.*]

In order to represent the complete semantic type of the noun *book*, we need to analyze its linguistic context in more detail. In this paper, we suggest its tentative dot-type, pointing out that phys_object·info is not specific enough to explain the linguistic behavior of the noun *book* and type coercion. Nevertheless, it is a crucial argument that the predicates taking *book* are derived from the dot-type of *book*. Based on the semantic types of *begin* and *book* and their distributional restriction, we argue about the type coercion phenomenon of *begin the book* in the next section.

4 Compositional Meaning and Type Coercion

Relying on the semantic type and selection restriction of *begin* and *book* given by their lexical meaning, they combine with each other by type coercion in the compositional process. Type coercion of begin-type verbs has some constraints. A part of the constraints were discussed in [PJPB1996]. Adopting their argument, we propose the constraints on type coercion of *begin the book*.

[6] In addition, there are different types of book such as an audio book, a video book, and an e-book. However, the distinction will be studied in the future work. we consider a book to represent a paper book here.

- **Constraints on type coercion of** *begin the book*
  - only in the control construction of *begin*;
  - when the subject is animate;
  - only when the missing predicate belongs to a process or an accomplishment type verb class;
  - and only when the missing predicate is a two-place verb which takes a subject and an object.

The first two constraints are explained in detail in [PJPB1996]. We focus on the last two in this paper. Given the list of governing verbs of *the book*, *begin* does not take the state or achievement type ones out of them. Consider the following examples.

(7) a. *John began **having** the book. (state)
b. *John began **losing** the book. (achievement)

The verb *have* cannot combine with *begin* because it is a state type and thus violates the semantic type restriction on its theme argument. In the same way, *losing* also cannot head the gerundive phrase, the object of *begin*. Therefore, these verbs are not candidates for the missing predicate in type coercion. The following examples show that type coercion is allowed when the head of the gerundive construction is accomplishment or process type.

(8) a. John began **reading** the book. (accomplishment)
b. John began (reading) the book.

In the above example, *reading the book* can combine with *begin* because it is an accomplishment eventuality. As shown in (8b), *reading* can be ellipsed and recovered.

Even when a gerundive satisfies the selection restriction of *begin* aspectually, it cannot be ellipsed if it is not a two-place predicate. See the following example:

(9) a. John began **giving** me books.
b. *John began me books.

Since *books* is plural, *giving me books* is a process aspectual type expression, although *give* is achievement type. Therefore, the sentence in (9a) is grammatical but *begin* does not allow the ellipsis of *giving*, as shown in (9b). This example shows that type coercion is allowed only when the missing predicate takes a subject and a direct object.

In the above sections, we explored which verbs can be the head of the gerundive construction with *the book* in the *begin* construction. However, there still is the problem of recovering an appropriate predicate in the interpretation of *begin the book*. The next section discusses it.

5 Interpretation: Ambiguity and Linguistic and Non-linguistc Context

Regarding type coercion, one of the biggest issues is to clarify how to recover the missing predicate in the interpretation of a type-coerced construction. As argued above, the constraints on type coercion exclude many predicates from the list of potential recoverable predicates. Nevertheless, there still remain many verbs as candidates. For instance, the following verbs can be recovered in the interpretation of *begin the book*:

- *read, write, publish, print, review, translate, promote, market, use,* etc.

In order to solve this issue, we suggest the distinction between linguistic and non-linguistic context. A linguistic context means the distributional behavior of an expression: that is, the other expressions co-ocurring with the expression in a sentence. On the other hand, a non-linguistic context is defined as the situation in which the utterance occurs.

The linguistic context of *begin the book* (e.g., the subject) can lead us to prefer a specific interpretation of the type coerced construction. Moreover, it sometimes excludes some predicates from the cadidate list for recovering the missing predicate. Consider the following example:

(10) a. **The author** began ({*writing*} / {*reading, promoting, translating, using, ...*}) the book.
b. The author began (?*{*writing, promoting, ...*} / {*reading, translating, using, ...*}) the **borrowed** book.

Because of the lexical meaning of *author*, the sentence in (10a) is predominantly interpreted as 'The author began *writing* the book'. However, the sentence is still ambiguous. The NP *the book* in (10a) can be interpreted as reading the book or other events, even though the subject of the sentence is *the author*. The NP *the borrowed book* in (10b) presupposes there exists the book. The presupposition implies that the author cannot write the book. The writer cannot promote or advertize the borrowed book on the purpose of selling it, either. Given that the linguistic context of *begin the book* narrows the range of candidates for the position of a missing predicate, its non-linguistic context, the utterance situation, finally disambiguates the sentence with *begin the book*. Consider the following examples:

(11) a. In the situation in which the author has been writing a new book,
The author began (**writing**) *the book.*
b. In the situation in which the author should read some books to prepare for writing,
The author began (**reading**) *the book.*

In the situation as in (11), it is natural that the missing predicates are recovered as *writing* and *reading*, respectively.

To sum up, both linguistic and non-linguistic contexts contribute to disambiguation in the interpretation process of type coercion construction. A type-coerced construction such as *begin the book* can have various interpretations including *reading* and *writing*, depending on linguistic and non-linguistic context. The disambiguation process of the type coercion construction goes from the lexical meaning of words to non-linguistic context (situation) via linguistic context (composition). Nevertheless, we prefer interpretation such as *reading* or *writing* to others in the interpretation of *begin the book*. Why do we prefer some interpretation to the others? In the next section, we introduce "conventional non-linguistic context" to answer the question.

6 Conventional Non-linguistic Context

Usually, the most preferable interpretation of *begin the book* is to begin reading the book, with no special non-linguistic context. We argue that it is because there is a "conventional non-linguistic context" linked to an expression which is commonly assumed by most normal people in a language culture. In other words, the lexical meaning of a word is a conventionalized meaning which assumes a conventional non-linguistic context linked to the word (cf. [GDnd]). In addition, the context holds in the compositional process of the words. For example, the noun *book* is conventionally linked to the situation of reading it, because reading is the most common activity which people do with a book. That argument is proved by the frequency of predicates combining with *the book* collected from various corpora. The verb *read* has the highest frequency and the second is *write* in most corpora. If we have no information about the non-linguistic context and the linguistic context is neutral, we usually interpret the sentence *John began the book* as *John began reading the book*.

However, the interpretation under the assumption of "conventional non-linguistic context" is only the most probable one (cf. [PC2015]). The ambiguity of *begin the book* is not dissolved yet. The specific non-linguistic context of the utterance finally disambiguates the type coerced construction *begin the book*. We mention here that the Qualia Structure in the GL is attractive in that it gives insight about the preferred interpretation of a type coercion construction by postulating telic and agentive qualia, in spite of the risk that qualia can be extended too much. The conventional activity of normal people with a book is closely related to the original function of the artifact and how to create it.

7 Conclusion

In this paper, we tried to rethink the condition which allows aspectual coercion of *begin* and the interpretation of the type-coerced construction (*begin the book*), introducing Asher's type-theoretical point-of-view and the "conventional non-linguistic context" based on Conventional Semantics ([GDnd]). The lexical semantic type of *begin* triggers some constraints on type coercion. In addition, we distinguished linguistic and non-linguistic context. Basically, the constraints

of aspectual coercion of *begin* are dependent on its lexical meaning. Regarding the interpretation of the type-coerced construction *begin the book*, we summarize our argument as follows:

I. **lexical meaning**
The semantic type and selection restriction of *begin* and *book* under the conventional non-linguistic context linked to the words

II. **compositional meaning**
The set of possible interpretations of the type-coerced construction *begin the book* with support of linguistic context
cf. *begin reading the book* is "preferred" as the interpretation of *begin the book* by its conventional non-linguistic context.

III. **contextual meaning**
The non-linguistic context finally dissolves the ambiguity of the type coerced construction *begin the book*.

A type-coerced construction is basically ambiguous until the non-linguistic context disambiguates it. We showed the three steps of interpreting a type-coerced construction above. Without any specific linguistic or non-linguistic context, conventionally-assumed situation linked to words works for disambiguation. Reading interpretation is the most probable one for *begin the book*, although it is still ambiguous.

There remain many important and interesting issues related with semantic type of words and type coercion. First, we need to define the concept of "conventional non-linguistic context" more strictly, considering its relation to the lexical meaning of words. Second, we need to decide what a dot-type is theoretically and propose the semantic type of *book* specific enough to explain its linguistic behavior (cf. [AN2011]). In the future research, we will explore the semantic type of words, including *book* and *begin*, and their semantic relation. In addition, we will develop the formalism of representing the interpretation mechanism which puts together lexical meaning and linguistic and non-linguistic context.

References

[AN2011] Asher, N.: Lexical Meaning in Context: A Web of Words. Cambridge: Cambridge University Press. (2011)

[GDnd] Gutzmann, D.: Semantics vs. Pragmatics L. Matthewson, C. Meier, H. Rullman, and E. Zimmerman, The Semantic Companion. (n.d.)

[IS2013] Im, Seohyun: The Generator of the Event Structure Lexicon (GESL): Automatic Annotation of Event Structure for Textual Inference Tasks. PhD Dissertation. Brandeis University. (2013)

[PC2015] Potts, C., Lassiter, D., Levy, R. and Frank, M. C.: Embedded Implicatures as Pragmatic Inferences under Compositional Lexical Uncertainty. Manuscript. (2015)

[PJ1995] Pustejovsky, J.: The Generative Lexicon. The MIT Press. (1995)

[PJms] Pustejovsky, J.: Lexical Semantics. Manuscript.

[PJPB1996] Pustejovsky, J. and P. Bouillon: Aspectual Coercion and Logical Polysemy. Pustejovsky, J. and B. Bogurave (eds.) Lexical Semantics: The Problem of Polysemy. Clarendon Press: Oxford. (1996)

Factivity and Presupposition in Dependent Type Semantics*

Ribeka Tanaka[1], Koji Mineshima[1,2], and Daisuke Bekki[1,2]

[1] Ochanomizu University
[2] CREST, Japan Science and Technology Agency
{tanaka.ribeka, bekki}@is.ocha.ac.jp, mineshima.koji@ocha.ac.jp

1 Introduction

Dependent Type Semantics (DTS, Bekki [3]) is a framework of natural language semantics based on dependent type theory (Martin-Löf [19]). In contrast to traditional model-theoretic semantics, DTS is a proof-theoretic semantics, where entailment relations are characterized as provability relations between semantic representations. One of the distinctive features of DTS, as compared to other type-theoretical frameworks, is that it is augmented with underspecified terms so as to provide a unified analysis of entailment, anaphora and presupposition from an inferential/computational perspective. In contrast to previous work on anaphora in dependent type theory (cf. Ranta [22]), DTS gives a fully compositional account of inferences involving anaphora (Bekki [3]). It is also extended to the analysis of modal subordination (Tanaka et al. [25]).

In this paper, we provide the framework of DTS with a mechanism to handle entailment and presupposition associated with factive verbs such as *know*. Although there are numerous studies on factive verbs in natural language semantics, they are usually based on model-theoretic approaches; it seems fair to say that there has been little attempt to formalize inferences with factivity from a computational and proof-theoretical perspective. On the other hand, various proof systems for knowledge and belief have been studied in the context of epistemic logic (cf. Meyer and van der Hoek [20]). However, such systems are mainly concerned with knowledge and belief themselves, not with how they are expressed in natural languages, nor with linguistic phenomena such as factivity presuppositions. Our study aims to fill this gap by providing a framework that explains entailments and presuppositions with factive verbs in dependent type theory.

2 Factive verbs and presupposition

We briefly summarize entailments and presuppositions triggered by factive verbs. Factive verbs like *know*, in contrast to non-factive verbs like *believe*, presuppose that the complement is true. There are two characteristic properties of presuppositions. First, a presupposition *projects* out of embedded contexts such as negation, question, and the antecedent of a conditional. Thus, not only (1) but also (3a-c) imply (2).

(1) John knows that he is successful.

* We thank the anonymous reviewers of TYTLES for helpful comments and suggestions.

(2) John is successful.

(3) a. John does not know that he is successful. NEGATION
b. Does John know that he is successful? QUESTION
c. If John knows that he is successful, ... CONDITIONAL

(4) a. If John is successful, he knows that he is.
b. John is successful, and he knows that he is.

Second, a presupposition is *filtered* when it occurs in contexts such as (4a, b). In general, if S' entails the presuppositions of S, constructions like S' and S and If S' then S do not inherit the presuppositions of S.

Unlike non-factive verbs, factive verbs can take interrogative complements as in (5), and license inferences as shown in (6) and (7) (Groenendijk and Stokhof [9]).[3]

(5) a. John {knows, *believes} whether Ann or Bob came.
b. John {knows, *believes} who came.

(6)
John knows whether Ann or Bob came.
Ann came.

John knows that Ann came.

(7)
John knows who came.
Ann came.

John knows that Ann came.

Interrogative complements themselves have presuppositions (Hintikka [12]; Karttunen [14]). For instance, it is natural to take *whether Ann or Bob came* as presupposing "Ann or Bob (but not both) came" and *who came* as presupposing "someone came". Figure 1 summarizes basic inference patterns for factive verb *know*.

E1	x knows whether A or B, $A \Rightarrow x$ knows that A
E2	x knows whether A or B, $B \Rightarrow x$ knows that B
E3	x knows who F, $F(a) \Rightarrow x$ knows that $F(a)$
P1	x knows that $P \rhd P$
P2	x knows who $F \rhd$ someone F
P3	x knows whether A or $B \rhd A$ or B (but not both)

Fig. 1. Entailments ($\Rightarrow$) and presuppositions ($\rhd$) associated with factive verbs.

3 Dependent Type Semantics

DTS (Bekki [3]) is a natural language semantics based on dependent type theory (Martin-Löf [19]). Since the work of Sundholm [24] and Ranta [22], dependent type theory has been applied to the analysis of various dynamic discourse phenomena, providing a type-theoretic alternative to model-theoretic frameworks such as DRT and Dynamic

[3] Here and henceforth, we take *whether A or B* as expressing an alternative question.

Semantics. Dependent type theory has also been applied to the study of natural language inferences in computational semantics with the help of modern proof assistants (Chatzikyriakidis and Luo [6]).

In dependent type theory, two type constructors Σ and Π play a crucial role in forming the semantic representations for natural language sentences. The type constructor Σ is a generalized form of the product type and behaves as an existential quantifier. An object of type $(\Sigma x : A)B(x)$ is a pair (m, n) such that m is of type A and n is of type $B(m)$. Conjunction $A \wedge B$ is a degenerate form of $(\Sigma x : A)B$ if x does not occur free in B. Σ-types are associated with projection functions π_1 and π_2 that are computed with the rules $\pi_1(m, n) = m$ and $\pi_2(m, n) = n$, respectively. The type constructor Π is a generalized form of the functional type and behaves as a universal quantifier. An object of type $(\Pi x : A)B(x)$ is a function f such that for any object a of type A, fa is an object of type $B(a)$. Implication $A \to B$ is a degenerate form of $(\Pi x : A)B$ if x does not occur free in B. See e.g., Martin-Löf [19] and Ranta [22] for more details.

Common nouns: types or predicates? There are two possible approaches to representing basic sentences like *A man entered* in dependent type theory. One is the approach proposed in Ranta [22] and Luo [17, 18], according to which common nouns like *man* are interpreted as *types* so that the sentence is represented as $(\Sigma x : \mathbf{man})\,\mathbf{enter}(x)$. A problem with this approach is that it is not straightforward to analyze sentences containing *predicate nominals*, such as (8a, b).

(8) a. John is *a man*.

b. Bob considers Mary *a genius*.

One might analyze (8a) as a judgement $\mathbf{john} : \mathbf{man}$; but then it is not clear how to account for the fact that such a sentence can be negated (*John is not a man*) or appear in the antecedent of a conditional (*If John is a man, ...*). One possible solution is to construe *be*-verbs as the so-called "*is*-of identity" along the Russell-Montague lines. Thus, (8a) is represented as $(\Sigma x : \mathbf{man})\,\mathbf{john} =_{\mathbf{man}} x$. This predicts that the predicate nominal *a man* introduces a discourse referent (in terms of Σ-types). However, contrary to this prediction, predicate nominals cannot serve as an antecedent of an anaphoric pronoun like *he* or *she* (Mikkelsen [21]); hence they do not introduce a standard discourse referent.[4]

As an alternative approach, we interpret a common noun as a predicate; thus *A man entered* is represented as $(\Sigma u : (\Sigma x : \mathbf{entity})\,\mathbf{man}\,x)\,\mathbf{enter}(\pi_1 u)$. This approach is in line with the traditional analysis of common nouns, so we can integrate standard assumptions in formal semantics into our framework. Note that although we do not take common nouns to be types, it is possible to refine type **entity** by introducing more fined-grained types such as ones representing animate/inanimate objects, physical/abstract objects, events/states, and so on (Asher [1]; Asher and Luo [2]; Bekki and Asher [4]; Retoré [23]). Such richer type structures will be needed to provide a proper treatment of lexical phenomena such as polysemy, coercion and, selection restriction

[4] See Fara [7] and Heim [11] for more discussion on the problems of the Russell-Montague analysis of predicate nominals.

	Π-types	Σ-types
Standard notation	$(\Pi x : A)B(x)$	$(\Sigma x : A)B(x)$
Notation in DTS	$(x{:}A) \to B(x)$	$\begin{bmatrix} x{:}\,A \\ B(x) \end{bmatrix}$
When $x \notin fv(B)$	$A \to B$	$\begin{bmatrix} A \\ B \end{bmatrix}$

Fig. 2. Notation for Π-types and Σ-types in DTS. $fv(B)$ means the set of free variables in B.

within the framework of dependent type theory. Our framework is consistent with such an extended type system.

In what follows, we will make use of the notation in DTS for Π-types and Σ-types as shown in Figure 2.

Presupposition in DTS. DTS is based on the paradigm of the Curry-Howard correspondence, according to which propositions are identified with *types*; the truth of a proposition is then defined as the existence of a proof (i.e., proof-term) of the proposition. In order to handle anaphora and presupposition in a compositional setting (see Bekki [3] for detail), DTS distinguishes two kinds of propositions, *static* and *dynamic* propositions. For any static proposition P, we say that P is true if P is *inhabited*, that is, there exists a proof-term t such that $t : P$. A *dynamic* proposition in DTS is a function mapping a proof c of a static proposition γ, a proposition representing the preceding discourse, to a static proposition. Such a proof term c is called a *local context*.

Underspecified term $@_i$ (where i is a natural number) is used to represent presupposition triggers. For instance, (9a) is represented as (9b), where definite article *the* introduces the term $@_1$ in the semantic representation.

(9) a. The apple is red.

b. $\lambda c.\, \mathbf{red}(\pi_1(@_1 c : \begin{bmatrix} x{:}\,\mathbf{entity} \\ \mathbf{apple}(x) \end{bmatrix}))$

The underspecified term $@_1$ is a function that takes a local context c as argument. A term of the form $@_i c : A$ is called *type annotation* and specifies that the term $@_i c$ has type A. In the case of (9b), the term $@_1 c$ is annotated with a type corresponding to the proposition *there is an apple* represented as a Σ-type. This means that the underspecified term $@_1$ takes a local context c as argument and returns a proof of that proposition. In this way, the annotated type represents the existential presupposition triggered by the definite description *the apple*.[5]

The type of an underspecified term $@_i$ can be specified by a type-checking algorithm (Bekki and Satoh [5]). Based on the inferred type, a proof search is carried out to

[5] Here we take it that the uniqueness presupposition is not part of the conventional meaning of a definite description but can be derived on pragmatic considerations along the lines of Heim [10]. Although it is possible to include the uniqueness presupposition in the type annotation A associated with *the*, the proof-search procedure to find the antecedent of an underspecified term would then become complicated.

construct a term of that type; then the underspecified term $@_i$ is replaced by the obtained term. This whole process corresponds to the process of presupposition resolution.

In the case of (9b), if a proof term for the existential proposition *there is an apple* is constructed given a local context c, it can substitute $@_1$. Such a proof construction is possible when (9a) appears in contexts such as *There is an apple and the apple is ...* and *If there is an apple, then the apple is ....* These cases correspond to the case of *presupposition filtering*. Note that given its type, $@_1 c$ is a pair of an entity and a proof of the entity being an apple. Accordingly, in (9b), the projection function π_1 is applied and returns the first element of the pair, i.e., an entity corresponding to the apple in question; then the predicate **red** takes this entity as argument.

When one cannot construct a proof required by $@_i$ from a given local context c, the existence of a term having the intended type can be assumed by means of the process of *accommodation*. The whole process of resolving an underspecified term $@_i$ is the same if the presupposition trigger is embedded under such a context as the scope of negation or the antecedent of a conditional. In this way, we can explain basic projection patterns of presuppositions.

4 Analyzing factivity in DTS

We will provide semantic representations for factive verbs by using underspecified term $@_i$ in DTS. For reasons of space, we will concentrate on the case of declarative complements and leave the analysis of interrogative complements for another occasion. We take the factive verb *know* as a representative case.

Declarative complements. We represent a sentence of the form a *knows that* P as $\lambda c.\, \mathbf{kn}(a, @_i c : Pc)$. The underspecified term $@_i$ here takes a local context c as argument and requires one to construct a proof term of type Pc, i.e., to find *evidence* of the (static) proposition Pc being true given the context c. Here P is a dynamic proposition expressed by the declarative complement of *know*. If such a proof term is constructed, it fills the second argument position of **kn**. Here predicate $\mathbf{kn}(x, t)$ can be read as *agent* x *obtains evidence* t. In sum, given a context c, the sentence x *knows that* P presupposes that there is a proof (evidence) of Pc and asserts that the agent x obtains it, i.e., x has a proof (evidence) of the proposition Pc. In the same way as the case of definite descriptions the analysis of presuppositional contents in terms of $@_i$ terms accounts for the projection and filtering properties of *know* as shown in (3) and (4).

The standard analysis of *know* in formal semantics follows Hintikka's [13] possible world semantics, which fails to capture the notion of evidence or justification that has been traditionally associated with the concept of knowledge. An advantage of dependent type theory is that it is equipped with proofs as first-class objects and thus enables us to analyze the factive verb *know* as a predicate over a proof (evidence) of a proposition.[6]

[6] The idea that a proof term serves as an antecedent of anaphor can be traced back to Ranta [22], where under the the assumption that proofs are identified with *events* it is claimed that aspectual verbs like *stop* presuppose the existence of a proof. See also Krahmer and Piwek [15],

In contrast to *know*, a verb like *believe* does not have factivity presupposition. Accordingly, we analyze non-factive attitude verbs like *believe* as a predicate over a proposition (cf. Tanaka et al. [25]). Thus, we treat factive and non-factive verbs as predicates having a different semantic type. This treatment of factive and non-factive verbs is consistent with Zeno Vendler's view that *know* and *believe* select different semantic objects, i.e., *know* selects a fact, while *believe* selects a proposition (Vendler [27]; Ginzburg [8]). Note that in our approach, the notion of facts is not taken as primitive but analyzed in terms of the notion of *evidence* of a proposition.

One advantage of this analysis is that it is consistent with, and directly applicable to, a language like Japanese in which factive and non-factive verbs require a different complementizer. In Japanese, there are two types of complementizer, *koto* and *to*. As Kuno [16] observed, factive verbs usually take a clause ending with *koto*, while non-factive verbs take a clause ending with *to*:

(10) John-wa Mary-ga kita *koto*-o sitteiru.
John-TOP Mary-NOM came COMP-ACC know.
'John knows (the fact) that Mary came.'

(11) John-wa Mary-ga kita *to* sinziteiru.
John-TOP Mary-NOM came COMP believe.
'John believes that Mary came.'

In general, *koto*-clauses trigger factive presupposition, while *to*-clauses do not. This contrast can be captured by assuming that a factive verb like *sitteiru* takes as its object a proof (evidence) of the proposition expressed by a *koto*-clause, while a non-factive verb selects a proposition denoted by a *to*-clause.

NP-complements. Our analysis can be naturally extended to factive verbs taking NP-complements. The factive verb *know* taking an NP-complement of the form *the N that A* shows different entailment patterns from non-factive verbs like *believe* and interrogative verbs like *ask* (Vendler [27]; Ginzburg [8]; Uegaki [26]): *know* does not license the entailment from x *Vs the rumor that* P to x *Vs that* P, nor that from x *Vs the question whether* A *or* B to x *Vs whether* A *or* B.

(12) a. John believes the rumor that Mary came. $\Rightarrow$ John believes that Mary came.
b. John knows the rumor that Mary came. $\not\Rightarrow$ John knows that Mary came.

(13) a. John asks the question whether Mary or Bob came. $\Rightarrow$ John asks whether Mary or Bob came.
b. John knows the question whether Ann or Bob came. $\not\Rightarrow$ John knows whether Mary or Bob came.

---

where it is briefly mentioned that the presuppositions triggered by noun phrases like *the fact that P* can be treated in a similar way. Although space limitations preclude us from examining these analyses in detail, our claim is that the idea that proofs act as antecedents of anaphora can best be applied to the presuppositions of factive verbs in general.

We take it that *know* is ambiguous between the *evidence*-taking reading (**kn**) and the so-called *acquaintance* reading. The latter is denoted as $\mathbf{kn}_{np}$. For example, x *knows the man* is represented as (14).

(14) $\mathbf{kn}_{np}(x, \pi_1(@_i c : \begin{bmatrix} x\text{: }\mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix}))$

(15) $\mathbf{kn}_{np}(x, \pi_1(@_i c : \begin{bmatrix} p\text{: type} \\ \begin{bmatrix} p = P \\ \mathbf{rumor}(P) \end{bmatrix} \end{bmatrix})$

Using the predicate $\mathbf{kn}_{np}$, we represent x *knows the rumor that* P as (15), which presupposes that there is a rumor whose content is identified with type P. When this presupposition is satisfied, (15) is provably equivalent to $\mathbf{kn}_{np}(x, P)$, which is clearly distinguished from the reading that x *has evidence of* P, hence, the non-entailment in (12b) follows. (12a), which contains *believe*, is represented in the same way as (15); thus it is equivalent to $\mathbf{believe}(x, P)$, hence we can derive the entailment in (12a). The asymmetry between *ask* and *know* in (13) can be explained in a same manner.

5 Conclusion

This paper analyzed entailments and presuppositions associated with factive verbs in the framework of DTS. We analyzed factive verbs as predicates taking a proof-object as argument, and non-factive verbs as predicates taking a proposition in the sense of dependent type theory. A fully compositional analysis of factive verbs in English and Japanese, including those with NP-complements, as well as an analysis of *wh*-complements, is left for another occasion.

References

1. Asher, N.: Lexical Meaning in Context: A Web of Words. Cambridge University Press (2011)
2. Asher, N., Luo, Z.: Formalisation of coercions in lexical semantics. In: Sinn und Bedeutung. vol. 17, pp. 63–80 (2012)
3. Bekki, D.: Representing anaphora with dependent types. In: Asher, N., Soloviev, S. (eds.) Logical Aspects of Computational Linguistics, Lecture Notes in Computer Science, vol. 8535, pp. 14–29. Springer (2014)
4. Bekki, D., Asher, N.: Logical polysemy and subtyping. In: Motomura, Y., Butler, A., Bekki, D. (eds.) New Frontiers in Artificial Intelligence, Lecture Notes in Computer Science, vol. 7856, pp. 17–24. Springer (2013)
5. Bekki, D., Satoh, M.: Calculating projections via type checking. In: Proceedings of TYTLES (to appear)
6. Chatzikyriakidis, S., Luo, Z.: Natural language inference in Coq. Journal of Logic, Language and Information 23(4), 441–480 (2014)
7. Fara, D.G.: Descriptions as predicates. Philosophical Studies 102(1), 1–42 (2001)
8. Ginzburg, J.: Resolving questions, I. Linguistics and Philosophy 18(5), 459–527 (1995)
9. Groenendijk, J., Stokhof, M.: Semantic analysis of *wh*-complements. Linguistics and Philosophy 5(2), 175–233 (1982)

10. Heim, I.: The Semantics of Definite and Indefinite Noun Phrases. Ph.D. thesis, University of Massachusetts, Amherst (1982)
11. Heim, I.: Definiteness and indefiniteness. In: Maienborn, C., von Heusinger, K., Portner, P. (eds.) Semantics: An International Handbook of Natural Language Meaning, vol. 2, pp. 996–1025. De Gruyter Mouton (2011)
12. Hintikka, J.: Knowledge and Belief. Cornell University Press (1962)
13. Hintikka, J.: Semantics for propositional attitudes. In: Models for Modalities, Synthese Library, vol. 23, pp. 87–111. Springer (1969)
14. Karttunen, L.: Syntax and semantics of questions. Linguistics and Philosophy 1(1), 3–44 (1977)
15. Krahmer, E., Piwek, P.: Presupposition projection as proof construction. In: Bunt, H., Muskens, R. (eds.) Computing Meanings: Current Issues in Computational Semantics. Kluwer Academic Publishers (1999)
16. Kuno, S.: The Structure of the Japanese Language. The MIT Press (1973)
17. Luo, Z.: Common nouns as types. In: Bchet, D., Dikovsky, A. (eds.) Logical Aspects of Computational Linguistics, Lecture Notes in Computer Science, vol. 7351, pp. 173–185. Springer (2012)
18. Luo, Z.: Formal semantics in modern type theories with coercive subtyping. Linguistics and Philosophy 35(6), 491–513 (2012)
19. Martin-Löf, P.: Intuitionistic Type Theory. Bibliopolis (1984)
20. Meyer, J.J., van der Hoek, W.: Epistemic Logic for AI and Computer Science. Cambridge University Press (2004)
21. Mikkelsen, L.: Specifying Who: On the Structure, Meaning, and Use of Specificational Copular Clauses. Ph.D. thesis, University of California, Santa Cruz (2004)
22. Ranta, A.: Type-theoretical grammar. Oxford University Press (1994)
23. Retoré, C.: The montagovian generative lexicon ΛTy_n: A type theoretical framework for natural language semantics. In: Proceedings of TYPES 2013. pp. 202–229 (2013)
24. Sundholm, G.: Proof theory and meaning. In: Gabbay, D., Guenthner, F. (eds.) Handbook of Philosophical Logic, vol. 3, pp. 471–506. Springer (1986)
25. Tanaka, R., Mineshima, K., Bekki, D.: Resolving modal anaphora in Dependent Type Semantics. In: Proceedings of LENLS11. pp. 43–56 (2014)
26. Uegaki, W.: Content nouns and the semantics of question-embedding. Journal of Semantics (to appear)
27. Vendler, Z.: Res Cogitans: An Essay in Rational Psychology. Cornell University Press (1972)

TYTLES summary

Robin COOPER (U. Gothenburg)
Christian RETORÉ (U. Montpellier)

Why types for lexical semantics?

- generative lexicon, coercion, learning, meaning in flux, dynamic system
- dependent types
- probability
- computability

Generative lexicon, . . . , dynamics

- types can remain constant over change of witnesses/inhabitants
- possibility of observing new witnesses of a type
- different from a Montagovian notion of sense: function from possible worlds to extensions — if you change the extension the sense changes
- the type a word is associated with can also change — structured types allow us to give an account of change not available in a Montagovian sense

Papers relating to some kind of dynamic aspect of types

- Stergios Chatzikyriakidis, Mathieu Lafourcade, Lionel Ramadier and Manel Zarrouk. Type Theories and Lexical Networks: Using Serious Games as the Basis for Multi-Sorted Typed Systems
- Staffan Larsson. Perceptual Meaning in TTR Judgement-based Semantics and Conceptual Spaces
- Simon Dobnik. Interfacing Language, Spatial Perception and Cognition in Type Theory with Records
- Ellen Breitholtz. Are Widows Always Wicked? Learning concepts through enthymematic reasoning
- Bruno Mery. The Relative Complexity of Constraints in Co-Predicative Utterances
- Livy Real and Alexandre Rademaker. An Overview on Portuguese Nominalisation
- Seohyun Im and Chungmin Lee. A Developed Analysis of Type Coercion Using Asher's TCL and Conventionality

Paper relating to the notion of structured types

- Frames in hybrid logic $\approx$ frames as record (types)
- can expressions of hybrid logic be thought of as record types (event types)?
- Laura Kallmeyer, Timm Lichte, Rainer Osswald, Sylvain Pogodalla and Christian Wurm. Quantification in Frame Semantics with Hybrid Logic

Dependent types

- story of Martin-Löf and donkey anaphora
- DRT conditionals sort of reinvented dependent types
- no corresponding general strategy in standard model theoretic semantics

Some uses of dependent types

- specific indefinites — Justyna Grudzinska and Marek Zawadowski. A Puzzle about Long-distance Indefinites and Dependent Type Semantics
- enthymemes/topoi — Ellen Breitholtz. Are Widows Always Wicked? Learning concepts through enthymematic reasoning
- presupposition — Daisuke Bekki and Miho Satoh. Calculating Projections via Type Checking; Ribeka Tanaka, Koji Mineshima and Daisuke Bekki. Factivity and Presupposition in Dependent Type Semantics

Probability

- probability associated with vagueness
- probability associated with type judgements *vs.* probability of possible worlds
- Peter Sutton and Hana Filip. Probabilistic Mereological TTR and the Mass/Count Distinction

Computability

- tractability more likely than with traditional possible world semantics
- Pepijn Kokke. Formalising type-logical grammars in Agda

Conclusion

- a broad span of approaches, different type theories
- but common assumptions and goals
- we should keep talking . . .