Abstract

This paper investigates two types of meaning shift in composition. The first examines the well-known shifts in meaning in predications, known as coercions, that involve aspectual verbs like start, begin or finish or a verb like enjoy in combination with non-event denoting direct object arguments. The second involves the modifications of meaning that result from the application of an adjective meaning to a common noun meaning, where the term coercion applies only to some of the shifts observed. Building on the framework and results of Asher (2011) and Asher and Luo (2012), my aim is to get a clearer picture of the different sorts of meaning shifts, what is responsible for meaning shifts and exactly how they affect content.

Keywords: Type composition logic; Coercion; Lexical semantics

1. Introduction

This paper investigates two types of meaning shift in composition. The first examines the well-known shifts in meaning in predications, known as coercions, that involve aspectual verbs like start, begin or finish or a verb like enjoy in combination with non-event denoting direct object arguments. The second involves the modifications of meaning that result from the application of an adjective meaning to a common noun meaning, where the term coercion applies only to some of the shifts observed. Building on the framework and results of Asher (2011) and Asher and Luo (2012), my aim is to get a clearer picture of the different sorts of meaning shifts, what is responsible for meaning shifts and exactly how they affect content.

What are coercions? Originally developed for use in simply typed programming languages (see, for example, Mitchell, 1983), coercions have been widely employed in linguistic semantics (Partee and Rooth, 1983; Pustejovsky, 1995; Pulman, 1997). Very roughly, a coercion is a function from one semantic value or one type to another that is employed when some problem arises in the construction of meaning. However, much of the linguistic literature on the subject has failed to provide a framework for analyzing coercions that is either formally or empirically adequate (including, for example, Pustejovsky, 1995), for reasons I rehearse in the next section.¹ I then present a simplified account of coercions from the Type Composition Logic of Asher (2011) with a brief critical discussion of an alternative provided in Modern Type Theory (Luo, 2010, 2011, 2012) and an improved treatment of some puzzling data in Asher and Luo (2012). In the final part of the paper, I turn to adjective noun composition and argue that the mechanisms of meaning shift are similar but different from those used for familiar coercions.

¹ See also Asher (2011), sections 2 and 3, for many more details.
2. Verbal coercions

2.1. The linguistic form of coercions

Coercions take place against a background of a theory of lexical meaning and meaning composition that takes selectional restrictions seriously. Normally a predication involving a predicate whose arguments do not meet its standard selectional restrictions will not result in a felicitous meaning. However, with coercion a predicate whose standard selectional restrictions are not met by its argument may still convey a coherent linguistic meaning because either one of the terms or the predication relation between predicate and argument is adjusted in some way so that the composition process may succeed.

In this paper, I am interested in coercions that involve incompatibilities between so called “simple” or “basic” types. What are simple types? Linguists working on word meaning developed a rich and complex typology of different sorts of entities that could affect semantic composition. Predicates come with sortal restrictions on their arguments; and if arguments meet those restrictions, then the predication is semantically felicitous. If they do not, then often the predications fail, as in (1):

(1) A prime number is soft.

The predicate is soft cannot felicitously apply to its argument a prime number, unless one of the terms is redefined or acquires a very idiosyncratic meaning in context. A natural explanatory hypothesis for the behavior of selectional restrictions is that selectional restrictions are type restrictions; the type of entity that satisfies the predicate is soft is a subtype of the type of entities that formal semanticists use (the type E), and this subtype is incompatible with the type assigned to numbers.\(^2\) Work in mathematical foundations and computer science uses strongly typed languages and a system of type checking or consistent type assignments to terms to assess the well-formedness of formulas or programs. Asher (2011) uses these tools in an analysis of semantic well-formedness, selectional restrictions and coercion, which I will follow here.

Let’s begin with what seems to be a very productive sort of coercion. In the following examples, we see two types of entities that a single expression may give rise to.

(2) a. John brought a bottle. It had a nice label/ It was yummy.
   b. John brought a bottle. It had a nice label and was yummy.
   c. John touched the bottle, which had been so yummy.

(2a,b) provide two different continuations for the first sentence, each containing pronouns that refer back to two different sorts of objects. If we analyze pronominal reference across sentences using Discourse Representation Theory or some other dynamic semantic formalism, we see something interesting. Depending on the continuation, one could infer that the first sentence of (2) makes available a discourse referent for the bottle or one for its contents that can be linked to the anaphoric pronoun in the continuation; but as (2bc) show, the first sentence makes available discourse referents for both the bottle and its contents.

Let’s now examine the examples in (2) from the perspective of selectional restrictions. Bottle intuitively types the entities satisfying the predicate as being of a type having to do with containers. Predicates like have a nice label apply to physical objects with stable surfaces, inter alia containers. So in the first continuation given in (2a), the pronoun picks up the discourse referent of the type CONTAINER. Assuming that anaphoric binding preserves type identity, then the discourse referent introduced by it that is the argument of have a nice label is of the right type to meet the selectional restrictions of the predicate. However, in the second continuation, the predicate is yummy requires its argument to be edible or drinkable, let’s assume. For simplicity, let’s assume that bottle is not in the type system a subtype of comestible foodstuff (though this is a simplification—the bottle might be made of chocolate). In that case, we have a case of coercion: in order for the predication to succeed, the predication must license the introduction of a discourse referent or variable that refers to the contents of the bottle and it is this referent that is the argument of the predicate is yummy. This discourse referent with the type CONTENTS satisfies the selectional restrictions of the predicate. Importantly, with this kind of coercion both the “coerced” denotation and the original denotation of the argument seem equally available.

There are other well-known examples of coercion—for example those involving aspechical verbs like start, begin and finish, as well as verbs like enjoy in English, where this is not the case. For instance, (3) is equivalent in meaning to (4):

(3) Julie enjoyed (started/finished) a book.
(4) Julie enjoyed doing something with (e.g., reading, writing, . . .) a book.

\(^2\) Asher (2011) argues for this thesis in detail.
enjoy requires an event as its direct object as in enjoy the spectacle, enjoy the view. This also happens when enjoy takes a free relative clause as its complement, as in enjoy (hearing) what he said. When the direct object of a transitive use of enjoy in a predication does not denote an event, the predication introduces some sort of eventuality that can serve to satisfy the selectional restrictions of the verb.

However, referring to the eventuality is often not very easy. We consider a predicate of events like last (for some amount of time) to be a good predictor of an event predication. We also are interested to see whether we can modify the event introduced with an adverb. It seems we can't.

(5) a. Jill started writing a book. The writing will last for years.
   b. Jill started a book. ?The writing /?That will last for years.
   c. *Jill started War and Peace, which will last for weeks.
   d. Jill started to a thorough reading of the article.
   e. Jill started a thorough article. (can't mean something like (5d))
   f. John is scared of starting War and Peace, because it will last for (will take him) weeks.
   g. John started War and Peace, but that won't last. He never gets far with long books.

This sort of data points to a difference between event coercions and container/contents coercions. In the latter, both a discourse referent for the container and one for the contents can be picked up by anaphors for subsequent predications; in the former, it seems that the event variable introduced to satisfy the selectional restrictions of aspctual verbs is not always available for anaphoric reference. Anaphoric reference to the coerced eventualities is not felicitous in (5b), in contrast to (5a). Further, the use of an appositive relative clause that is supposed to modify the event introduced in the coercion is not, grammatical, and neither is adjectival or adverbial event modification in (5e). Non-restrictive relative clauses are only licensed if the DP they modify denotes an entity of the sort the relative clause can apply to. Potts (2005). Examples like (5c,d) show that the coerced eventuality is neither available for coreference nor modification, indicating that it is not present in our semantic logical forms.

Surprisingly, anaphoric reference to the eventuality becomes better when certain discourse patterns are introduced, in particular causal relations like Explanation (5f) and Contrast relations (5g), but not in cases where we are continuing a topic or in an Elaboration (5b,c). Some coercions behave even more bizarrely. In these examples from Asher and Luo (2012), the same discourse sensitivity is observed as in the above examples, but the original denotation of the argument seems unavailable except in the aforementioned discourse patterns.

(6) a. ??The omelette left without paying, although it was very yummy.
   b. ??The omelette, which has fresh mushrooms in it, left without paying.
   c. The omelette that had fresh mushrooms in it left without paying; the omelette with ham and cheese paid.3
   d. ? The delicious omelette has left without paying.
   e. The omelette has ordered. He wants it with mushrooms.
   f. The omelette is getting restless. But you can tell him it's almost ready.
   g. The omelette left without paying because he found it disgusting.
   h. The omelette is enjoying it.
   i. *The omelette ate it with gusto.

The impossibility of using non restrictive relative clauses (6b) or even certain non restrictive adjectival modifications that require the DP to have its standard meaning (6d) is surprising, especially in contrast with (6c) with the restrictive relative clause. Aspctual verbs and enjoy allow non restrictive modification of the standard meaning of their arguments; e.g., Julie enjoyed the novel, which is a thrilling murder mystery. According to the evidence, the omelette (6a,b,d) simply doesn't refer to the food stuff at all. Nevertheless, with suitable discourse configurations involving particular discourse relations, as in (6e–g), anaphoric reference to the foodstuff, the original meaning of the omelette is felicitous. (6h) is particularly puzzling; it's fine in comparison with (6i), which is totally infelicitous.

We must do several things to apply type coercions to examples like (3) or (2). First, we must extend the standard type system used in Montague Grammar, which includes only two atomic types e (the type of entity) and t (the type of truth values), “downwards” to include appropriate subtypes of e. We have to have subtypes for container and contents and individual physical objects and eventualities as inputs to the type shifting mechanism, whatever form this may take. Second, if we take, as we do, coercions to be licensed by conflicts between the requirements of predicates like be yummy or have a nice label, we have to assign these predicates types or lexical meanings such that they require one of these types as arguments.

---

3 Thanks to a reviewer for this example.
Characterizing precisely what the subtyping relation is in a system with many subtypes of $e$ constitutes a second difficulty for any satisfactory account of coercion. According to the standard logical system underlying Montague Grammar, the subtyping relation of $\subseteq$ is understood as subset (Church, 1940), with the accompanying subsumption rule: if $A$ is a subtype of $B$, then every object of type $A$ is of type $B$. If atomic types are understood as sets and types of predicates are understood as functions from the objects of the domain type to truth values, we run into a severe problem. For instance, suppose we have a domain type of physical objects $P$, then the first order properties of physical objects are of type $P \rightarrow \tau$. How do these properties relate to the general type of first order properties, that is the properties whose domain type is just the set of all entities $e$? Grammar and common sense tells us that the physical first order properties should be a subset of the general type of first order properties, just as physical objects are a subtype of the general type of objects $e$. But in standard type systems the sets $P \rightarrow \tau$, the type of all functions from physical objects to truth values, and $e \rightarrow \tau$ have no common elements. So the types are not related in any way. We cannot extend the subtyping hierarchy from the atomic types to the predicates in any obvious way. This is a serious problem because it means if we make $\textit{man}$ have type from $\textit{man} \rightarrow \tau$, it won't be able to combine with the type of a higher order variable of type $e \rightarrow \tau$. The entire compositional calculus of traditional Montague Grammar collapses.

A final problem for any theory of coercion is the core of the problem: how do we make a predication that would otherwise violate selectional restrictions and so be semantically anomalous introduce the right sort of argument for the predicate so that we predict successful predications in the coercion examples? There are three options for handling the phenomena with type coercion, and only one of them is viable, as argued in Asher (2011).

The first possibility is to assume that the mechanism of type coercion transforms the denotation of the noun phrase $\textit{the book}$ into some sort of eventuality denoting expression. If that is the case, then how can we access in subsequent discourse the referent of $\textit{the book}$?

(7) Julie enjoyed a book. It was a mystery.

These familiar observations show that we cannot shift the meaning of $\textit{the book}$ to some sort of eventuality. Or at least if this happens, whatever process is responsible for the shift must also allow $\textit{the book}$ to retain its original, lexical meaning and its original contribution to logical form.

In addition, coercions do not affect the whole of the DP argument or at least not in all cases of coercion. This contrasts with the behavior of predications involving so called dual aspect nouns like $\textit{book}$ or $\textit{lunch}$. These nouns are called dual aspect nouns because they appear to have simultaneously two incompatible types—abstract object and physical object for $\textit{book}$ and food and event for $\textit{lunch}$ (Asher, 2011). They have an inherent kind of polymorphism. When a verb selects one of their types, we note a change in the quantificational domain of the DP. Consider the following examples from Asher (2011).

(8) a. John has mastered every mathematics book in the library. (Here the quantification is over abstract objects. John has mastered the contents, and may never have been to the library).
   b. John has stolen every mathematics book in the library (Here the quantification has to be over physical volumes; John has to have stolen, say, all the copies of Rudin's $\textit{Real and Complex Variable Analysis}$).
   c. $\textit{The best of Robert Louis Stevenson}$ is three great books in one (here we have a quantification over abstract objects in the first DP and over physical volumes in the DP with the N' ellipsis).

Shifts in the whole interpretation of the DP, however, don't occur with most coercions. Consider

(9) George enjoyed started many books last weekend.

In (9), the quantification over books remains a quantification over books and not over eventualities. That is, (9) does not have a reading on which there are multiple events of George's enjoying just one book, which should be possible if coercions licensed quantificational shifts in the way that dual aspect nouns do. Thus, whatever coercion does, it makes its adjustments locally to the predication relation in logical form and does not affect the DP interpretation or quantificational domain.

The second alternative is to shift the predicate. In (3) $\textit{enjoy}$ should mean something like $\textit{enjoyed doing something with}$. This alternative is imprecisely stated, but if it means that the actual coercing verb should shift its content to one whose

---

4 For more details see Asher (2011).
5 The account in Pustejovsky (1995) seems to adopt this view.
6 As one reviewer notes, there may be exceptions involving "omelette" type coercions: $\textit{The two omelettes paid}.
7 Nunberg (1995), and others have pursued this line of attack.
selectional restrictions allow it to combine with its argument, then this proposal fails as well. Constructions using ellipses are instructive. The meaning of the activity is predicted on the predicate modification view to shift with the choice direct object during semantic composition. But if that is the case, then the ellipsis predicts a peculiar reading of the second clause of (10a) where I have a car that is parked out back and Mary’s car has a car that is parked out back. Consider

10  a. I’m parked out back and Mary’s car is too.
    b. Julie enjoyed her book and Isabel her parade.
    c. Julie enjoyed (reading) a book and Isabel (watching) a movie.
    d. Julie enjoyed a book and watching a movie.
    e. I really enjoyed the book and how it uses someone’s life in the story and gives ideas on how to stay unstressed and calm. (http://www.silverhillsontheroad.com/teens/teenbook)

Let’s look at the gapping examples (10b,c) in a bit more detail. At the level of syntax, I assume a coindexing between the verb in the first clause and the elided site in the second. The natural semantic interpretation of this is identity. But then the view that the verb itself shifts meaning leads to crazy predictions: Isabel has to enjoy doing something to her parade, the same sort of thing that Julie enjoyed doing to her book. And thus, we can’t get the natural interpretation of (10c) at all. (10c,d) are also trouble for the predicate modification view. To handle the DP coordination, the most sensible option is to resort to a distributive reading of the coordinated DP, according to which (10d) is analyzed as

(10d’) Julie enjoyed a book and [enjoyed] watching a movie.

At the syntactic level, we might assume that the link between the overt occurrence of enjoy and the deleted occurrence, [enjoyed] is once again a matter of coindexing. This leads to the difficulty we already saw with gapping: the predicate modification view predicts that Julie enjoyed doing something to watching a movie, which is uninterpretable. All this is strong evidence that coercion does not involve a semantic adjustment to the meaning of the coercing predicate at the moment of composition.

Since we’ve seen that you can’t simply shift the meaning of the predicate and you can’t shift the meaning of the argument, what is left? The answer from TCL (Asher, 2011) but also (Egg, 2003: Dölling, 2003) is that you change the relation of predication that holds between the predicate and the argument, though TCL goes into much more detail as to how subtyping and higher order types are to be interpreted and as a result shifts the predication relation more locally in comparison to these approaches. A clash between the type of the argument and the type presupposition of the predicate induces, not a type shift in either the argument or the predicate, but rather a type shift on the predication relation itself. TCL implements this by introducing a functor that is inserted around the core part of the verb or predicate meaning so that it can apply to the variable introduced by the argument. This very local modification of the predication relation is needed, as we’ll see, because in TCL the full verb meaning lies at the root of the meaning composition, and an application of the functor to the full verb meaning would affect the verb’s arguments. Thus, the meaning of the argument does not shift—it remains what it always was; the predicate also retains its original meaning and can be recovered for gapping or sluicing examples such as (10a). What changes is the “glue” that links them together.

To sum up, the evidence suggests that coercion is a phenomenon that has to be handled within the mechanism of meaning composition and that a proper treatment of coercion requires a rethinking of how predication works. Both Asher (2011) and Luo (2010, 2012) work out this idea in different ways, which I now detail.

2.2. Coercions in TCL

Developed in Asher (2011),8 Type Composition Logic (TCL) integrates on top of a standard intensional semantics for English a system of types with a category theoretic or proof theoretic interpretation. A lexical entry in TCL thus conveys two levels of meaning—one considering the type requirements or selectional restrictions that an expression imposes on its arguments and a consequent typing of the term itself, and the other a standard denotational meaning.

TCL develops a solution to the problem of subtyping while keeping the subtyping relation subsumptive. Informally, this means that if A is a subtype of B, it is always a subtype of B. In TCL, subtyping is a matter of limited deduction. TCL’s basic subtypes of e are types of individuals and verify intuitive subtyping relations like the fact that Man is a subtype of Animal.

A common noun like man has in TCL like in Montague Grammar a translation into a term of higher order intensional logic with a lambda bound individual variable x with Mar(x) as a sub-formula. The type assigned to x is not the type Man in TCL but rather Physical-Object, in order to allow for a distinction between semantic anomaly and semantic falsity. Thus, in TCL that’s a man said while pointing to a statue is false whereas the set { {}, { {} } } is a man is semantically anomalous, since

---

8 But with precursors in Asher (2007).
the type on the individual level variable of the DP (INFORMATIONAL OBJECT) does not match that of the type presupposition of man.

To carry through intuitive subtyping relations for higher order types, TCL gives first order property variables of the logic a different type from that in Montague Grammar; they have a type that involves a bounded existential quantification over all subtypes of e of the form \( \exists x \subseteq e(x \rightarrow \top) \). Informally, this says that the type of a property is the disjoint sum of all functional types whose first argument is a subtype of e into \( \top \). Since man has (roughly) the type \( P \rightarrow \top \) and \( P \) is a subtype of \( e \), we can easily derive that the type of \( \lambda x \text{man}(x) \) is a subtype of the type of first order properties, thus allowing such terms to combine with a determiner of the form \( \lambda P \text{Q}(\exists x(P x \land Q x)) \). We can also prove that the type of physical first order properties, \( \exists x \subseteq \top(x \rightarrow \top) \), the first order properties, as desired, assuming that \( \subseteq \) is transitive and obeys the intuitive subtyping relations among atomic or simple types.

In TCL types are used to guide the construction of logical form and are treated as a kind of presupposed information. Selectional restrictions of predicates are like presuppositions, because they must be satisfied somehow or else the predication will be anomalous and difficult to assign truth conditions to. (1) is an example where the type presuppositions of the predicate can’t be satisfied. They are also not part of ordinary “at issue” content because they escape the scope of operators like negation and question operators, as Asher (2011) details. The second point of comparison between selectional restrictions and presuppositions is that like presuppositions, selectional restrictions may be satisfied in various ways. When a predicate requires that an argument have a type \( \tau \), then as long as its argument type \( \sigma \) is compatible with \( \tau \) (i.e. \( \sigma \cap \tau \neq \bot \)), type checking succeeds and the predicate and the argument combine together in TCL with the argument taking as type the meet \( \cap \) of \( \sigma \) and \( \tau \). But other forms of type presupposition satisfaction are possible, for instance, in cases of coercion.

TCL does exploit the fine-grained types in adjectival noun composition and adjustments for type presupposition satisfaction. For man, the finest-grained type associated with the lambda bound variable \( x \) in the lexical entry for man is man, but it doesn’t enter into the type of the lexical entry for the reasons I have explained. These most specific types are represented in Asher (2011) via a special function called \( \text{HD} \) or head type on terms that returns the most specific type associated with the variable in the term that will combine with other elements. Thus the head type of the entry \( \lambda x \text{man}(x) \) would be man. It is these types that guide the type adjustment mechanism for coercion.

With coercions like (2) or (3), repeated below, there is no common subtype in the type hierarchy for the type presupposition of the predicate and the argument; the contents of a bottle is of a type incompatible with that of the bottle that contains it, and similarly, the eventuality required by \( \text{start} \) or \( \text{enjoy} \) is a distinct and incompatible type with that of a physical or abstract object like a book—these types have different identity and individuation criteria.

(2) a. John brought a bottle. It had a nice label/ It was yummy.
   b. John brought a bottle. It had a nice label and was yummy.
   c. John touched the bottle, which had been so yummy.

(3) Julie enjoyed (started/finished) a book.

However, predicates like \( \text{start} \) license a justification of their type requirements by allowing a type transformation of its object argument. This transformation, however, is not a simple function on the linguistic argument; it introduces material linking the actual argument of the verb to an object of the desired type in a manner akin to the sort of presupposition justification employed in bridging. This material for instance will introduce an event that is linked to an actual non event-type argument of \( \text{start} \). The type of this event is polymorphic; it is a function of the type of the object argument as well as the subject argument of the verb. This changes the relation between the predicate and its arguments and has a truth conditional content. The transformation also has a narrow scope, operating only on the variable introduced by the DP into the verb’s argument. Otherwise, the functor could affect the semantic contribution of the DP, which we do not want.

So for example (3), the transformation licenses the introduction of a functor that applies to the verb’s entry, allowing to the result to combine with the original argument of the verb. In TCL, the functor applies to the core predicate provided by the verb, which is of the form \( \lambda x \lambda y \text{enjoy}(y, x, e) \), because that is where the type conflict is detected:

\[
\lambda P \lambda u \lambda e' \lambda v (\exists e' (P (\lambda v(e))(e') \land \phi(e, v)))
\]

Note that the functor introduces into logic a description of an eventuality as the second argument of \( \text{enjoy} \), which will go in for the \( \lambda \) bound variable \( P \); note also that this description is itself underspecified, because without a particular context, we

---

I have adopted a Neo-Davidsonian construal of verbal entries as these are more familiar than TCL’s approach. The differences don’t matter for the purposes of this paper.
In (14a--c), the subject DP provides for a different reading of the coercion. This is another thing that (Egg, 2003; Dölling, 2003) do not take into account.

We assume that type requirement on its object that is complex but which I type coercion for its direct object argument is signaled in the type requirements associated with this second argument by the λ"rules of the P of the head type of the subject DP. Nevertheless, the entry for the verb itself does not have the complex type associated (the type of objects that the DP quantifies over or denotes) to some eventuality type that is a function of this head type and between the type requirements of the verb DP denotations (which are generalized quantifiers) to DP denotations to typing contexts to propositions. This dissociation with a function from agents to eventualities to propositions (or truth values); it has the type of being a function from general adjustments in the predication relation at various points in the composition process.

To illustrate the TCL approach to event coercing predicates, let’s look at the lexical entry for enjoy. In TCL a transitive verb takes two generalized quantifier arguments. To pass type requirements of the verb to the arguments, TCL uses the method of continuations (de Groote, 2006); the typing context is made part of the lexical entry and is labeled by the lambda bound variable π. π is a list of type requirements provided by the context; a verb simply adds its type requirements to the list and passes that list to the DP where the typing is checked for coherence. I’ll simplify here the TCL entry and only put the type requirements of the verb on its second or internal, direct object argument, ignoring the effects of the predicate on the subject argument.

(13) \[ \lambda \Psi \lambda e \lambda \Phi \lambda \Pi \Phi(\pi) (\lambda v \Psi(v) (\lambda y_1 \lambda \Pi_1 \lambda P(v, y_1, e', \pi_1 + \text{EVT} - \epsilon(\text{HD}(\Phi), \text{HD}(\Psi)))))) \]

This says that the object DP must contribute a second argument of eventuality or EVT type.\(^{10}\) The fact that enjoy licenses a coercion for its direct object argument is signaled in the type requirements associated with this second argument by the type \(\epsilon(\text{HD}(\Phi), \text{HD}(\Psi))\). This last type signals that enjoy permits a natural transformation from the head type of the object DP (the type of objects that the DP quantifies over or denotes) to some eventuality type that is a function of this head type and of the head type of the subject DP.\(^{11}\) Nevertheless, the entry for the verb itself does not have the complex type associated with a function from agents to eventualities to propositions (or truth values); it has the type of being a function from general DP denotations (which are generalized quantifiers) to DP denotations to typing contexts to propositions. This dissociation between the type requirements of the verb’s arguments and the type of the term allows us to apply type justifications and adjustments in the predication relation at various points in the composition process.

To see how to derive the logical form in (12), let’s apply this lexical entry in (13) to the arguments it is supplied with in (3). We assume that Julie has the usual entry for a name in Montague Grammar inter alia. The DP a book has the entry with a type requirement on its object that is complex but which I’ll just abbreviate as BOOK:

(15) \[ \lambda P \lambda \pi \exists x(\text{book}(x, \pi + \text{BOOK}) \land P(x)(\pi + \text{BOOK})) \]

where \(P\) ranges over first order properties. Thus, the \(\lambda\) term for the book will combine with a sub formula of the verb’s lexical entry, \(\lambda y_1 \lambda \Pi_1 \lambda P(v, y_1, e', \pi_1 + \text{EVT} - \epsilon(\text{HD}(\Phi), \text{HD}(\Psi)))\), so that we can pass the type requirements of the DP to the “core” verbal content and check for type well-formedness there. Combining these two entries together, and using the rules of the \(\lambda\) calculus, we can simplify the resulting formula to a certain degree:

(16) \[ \lambda \Phi \lambda e \lambda \Phi(\pi) (\lambda v \exists x(\text{book}(x, \pi + \text{BOOK}) \land \lambda y_1 \lambda P(v, y_1, e', \pi + \text{EVT} - \epsilon(\text{HD}(\Phi), \text{HD}(\Psi))) + \text{BOOK})(x))) \]

\(^{10}\) enjoy is a control verb and imposes that the agent of the eventuality is the subject of enjoy, but we skip this detail here.

\(^{11}\) This is needed to deal with examples like those in Asher (2011) that show that the subject can often affect readings of coercions.

(14) a. The cleaners have started on the living room.
    b. The painters have started on the living room.
    c. The exterminators have started on the living room.

In (14a--c), the subject DP provides for a different reading of the coercion. This is another thing that (Egg, 2003; Dölling, 2003) do not take into account.

Please cite this article in press as: Asher, N., Types, meanings and coercions in lexical semantics. Lingua (2015), http://dx.doi.org/10.1016/j.lingua.2015.01.001
Here the normalization (elimination of the λ bound variables) fails to go further, because of the type mismatch on \( x \), the internal argument of \( \textit{enjoy} \). Books are not events, at least literally speaking. But because \( \textit{enjoy} \) permits the introduction of a functor over its lexical entry in (13), we can eliminate this type clash. This functor takes (13) and applies the lambda term in (18) to the core λ abstract given by the verb, \( \lambda y \textit{enjoy}(v, y, e', \pi' + \texttt{EVT} - \varepsilon(\texttt{HD}(\Phi), \texttt{HD}(\Psi)) + x : \texttt{BOOK}) \), where the inconsistency is detected.

How does it do this? In Asher (2011) a specific rule is used to introduce the functor on a component of logical form. While we need a rule to introduce the functor into the derivation, we can simply use the rules of the \( \lambda \) calculus to complete the derivation. Notice that in fact (16) is equivalent to

\[
\lambda \chi \lambda \Phi \lambda e \lambda \pi \pi(\pi'(\pi' + e) + x : \texttt{BOOK}) \cap \\
\lambda \chi (\pi + \texttt{K} - \varepsilon(\texttt{HD}(\Phi), \texttt{HD}(\Psi)) + x : \texttt{BOOK})))(\lambda \pi' \lambda y \textit{enjoy}(v, y, e', \pi'))
\]

Our functor in (18) takes (17) as input and returns it unchanged except that the \( \lambda \) term in (18) is applied to the second argument. Note that (18) introduces a variable of the right type that can satisfy the type requirements of \( \textit{enjoy} \) on its direct object argument.

\[
\lambda P \lambda u \lambda \pi''(\exists z \texttt{P}(z)(\pi'' + z : e(\texttt{HD}(\Phi), \texttt{BOOK})) \cap \\
\phi_{(\texttt{HD}(\Phi), \texttt{BOOK})}(z, v, u, \pi'' + z : e(\texttt{HD}(\Phi), \texttt{BOOK})))))
\]

This allows us to now to repair the type conflict in (16) and hence to reduce (17) to a formula with the desired, underspecified meaning:

\[
\lambda e' \lambda \Phi \lambda \pi \pi(\pi' + \texttt{AG}) (\lambda v \exists x (\texttt{book}(x, \pi + \texttt{BOOK}) \cap \\
\exists z \textit{enjoy}(v, z, e', \pi) \cap \phi_{(\texttt{HD}(\Phi), \texttt{BOOK})}(z, v, x, \pi)))))
\]

The functor I’ve highlighted here suffices to handle event coercion with verbs whose type presuppositions are sensitive to both the type of the subject and object. There are many other functors involved in different forms of coercion. The underspecification in the functor in (18) allows TCL to capture the observation that even with ellipsis, we can get different coercions. Consider the following example:

(20) Julie enjoyed her book and Isabel did too, though Julie enjoyed proofreading her new semantics book, while Isabel enjoyed reading her story book.

Because the VP is copied with the underspecified coercion, different specifications can occur, depending on the subject that combines with the verb. To specify the underspecified formula to the preferred reading, TCL relies on contextual rules and on defeasible generalizations about specifications. For example, in the absence of other information, we might specify \( \phi_{(\texttt{HD}(\Phi), \texttt{BOOK})}(j, x, \pi) \) to \( \textit{read}(j, x, \pi) \). Note in addition that the coercion only affects the way the verb meaning connects to its arguments; thus, TCL predicts that the quantificational domains of the arguments of a coercing predicate won’t shift in the coerced prediction—another important prediction.

2.3. TCL and coercions in Modern Type Theory with coercive subtyping

Modern Type Theories (MTTs)\(^{12}\) can be used as foundational languages for formal semantics.\(^{13}\) These frameworks emulate a denotational semantic framework like Montague grammar (Montague, 1974), although MTTs provide arguably better alternatives because of their rich type structures that are lacking in the Montague semantics. The representations are very simple because MTT is designed for the first order set theoretic framework of mathematics, not for the general higher order intensional logic framework that is the \textit{lingua franca} of formal semanticists.\(^{14}\)

One important difference between MTT approach and other formal approaches like TCL is that common nouns (CNs) are interpreted as \textit{types} rather than predicates. For instance, common nouns like \textit{table} and \textit{man} are interpreted as types \texttt{TABLE} and \texttt{MAN}, rather than as predicates of type \texttt{E \rightarrow PROP}, where \texttt{PROP} is the type of propositions. In MTT, each common noun has the most specific atomic type associated with the noun in TCL as its principal semantic contribution; common

\(^{12}\) Examples of modern type theories include Martin-Löf’s type theory (MLTT) (Martin-Löf, 1984; Nordström et al., 1990), the Unifying Theory of dependent Types (UTT) (Luo, 1994) and the type theory implemented in the Coq proof assistant (pCIC) (The Coq Development Team, 2010).

\(^{13}\) Ranta (1994) did this for Martin-Löf’s type theory.

\(^{14}\) Though given MTT’s powerful background language, it can be adapted to a more general setting pace the difficulties I note below.
nouns have a simple type. This difference allows MTT approaches to avoid certain complexities of TCL like the use of continuations to pass type information. On the other hand, this means MTT does not distinguish between semantic anomaly and falsity, whereas this is a core concern of TCL. Because there is no difference in MTT between type presuppositions and fine grained associated types, common nouns cannot be predicates without threatening intuitive subtyping relations. Were man to introduce a predicate, say of type MAN → PROP, all the intuitive subtype relations would go the wrong way—since ANIMAL → T is a subtype of MAN → PROP, if we assume that MAN is a subtype of ANIMAL. For another it means that determiners are not relations between sets or intensions but rather functions from atomic types to predicates to propositions, which complicates the treatment of generalized quantifiers. Another problem is how to deal with modified common nouns as arguments to determiners. Adjectives are interpreted as predicates whose domains are not the type of all entities, but the types over which the adjectives are meaningful. For instance, ‘handsome’ is interpreted as a predicate of type MAN → PROP. This causes trouble for a uniform treatment of determiners.

As in TCL, in MTT a reexamination of subtyping becomes essential since subtypes are crucially needed even for basic semantic constructions. Luo (2010, 2012) propose coercive subtyping as a subtyping framework. An MTT with coercive subtyping is a powerful language with rich type structures which provide a variety of useful mechanisms for formal semantics (Luo, 2011, 2012). Coercive subtyping provides us a framework to interpret various linguistic coercions. The basic coercive subtyping mechanism also supports type-shifting in predicates. Coercive subtyping as formulated in a logical framework coerces f(a) into fc(a) by inserting the coercion c into a gap between f and a (Luo, 1999).

To see how coercive subtyping works, let’s work through an example where we concentrate just on the first order logical forms that result. Let’s reconsider (3), repeated here as (21):

(21) Julie enjoyed a book.

In MTT, a first order formal interpretation of (21) is (22):

(22) \( \exists x : \text{book} \cdot [\text{enjoy}] (j, x) \)

where

(23) \([\text{enjoy}] : \text{EVENT} \rightarrow \text{HUMAN} \rightarrow \text{PROP} \).

However, the domain type of \([\text{enjoy}] (j)\) is Event, which is different from Book! Then, how can \([\text{enjoy}] (j, x)\) in (22) be well-typed? The answer is that, in the framework of coercive subtyping and, in particular, under the assumption of the following coercion:

(24) \( \text{BOOK} \ll_{\text{reading}} \text{EVENT} \)

\([\text{enjoy}] (j, x)\) is coerced into (and, formally, equal to) \([\text{enjoy}] (j, \text{reading}(x))\) and hence well-typed. Informally, the sentence (21) is coerced into (25):


This linguistic use of coercion in MTT, which (Luo, 2011, 2012; Asher and Luo, 2012) develop, looks like a version of argument type shifting. But there is an important difference: the coercion works very locally. That is, we can operate just on the variable introduced by the DP a book before it combines with the verb—as in TCL. So the whole DP does not shift in type. In effect, the coercion rule is the image in MTT of the functor application rule of TCL alluded to above.

Furthermore, coercions here are local in MTT, because they have a scope; only one occurrence x in the logical form in (22) is considered as an eventuality, the one within the scope of c. In MTT, we do not have the destructive effects of type shifting with subsumptive subtyping; with coercive subtyping in MTT both the original meaning and the shifted meaning are available for subsequent continuations and anaphoric coreference, as the coercion can be applied multiple times. Coercions in MTT do not thus introduce new terms; rather, an occurrence of a term is in a certain context interpreted

15 Coercive subtyping was studied in 1990s (Luo, 1999) for applications of MTTs to, for example, formalization of mathematics and verification of programs, as supported by proof assistants such as Coq (Saïbi, 1997), Lego (Luo and Pollack, 1992; Bailey, 1999), Matita (The Matita proof assistant, 2008) and Plastic (Callaghan and Luo, 2001). It is also worth noting that the traditional notion of subtyping, subsumptive subtyping, is not adequate for MTTs. Coercive subtyping is. The coercive subtyping extension is conservative, which implies that the consistency of the original type theory is preserved.

Please cite this article in press as: Asher, N., Types, meanings and coercions in lexical semantics. Lingua (2015), http://dx.doi.org/10.1016/j.lingua.2015.01.001
differently—viz. reading(x) vs. x. In some sense this is a simpler solution than TCL’s. In TCL the introduction of a separate term for the eventuality is essential because of subsumptive subtyping.

However, this simplicity comes at a price, which shows up in the difference between TCL’s and MTT’s approach to ambiguity. Note that, in the above, we have considered only one possible coercion in (24): from ‘enjoy a book’ to ‘enjoy reading a book’. As we noted in the previous section, however, there are in fact context-dependent ‘multiple coercions’: e.g., (21) could have meant ‘Julie enjoyed writing a book’ (cf., (3)); there could also be several reading events of that book. Coercive subtyping requires contextual uniqueness of coercions; any two coercions between the same two types (in the same context) must be the same Luo (1999).

Thus, if we want to use multiple coercions, we must explicitly mark the context where the coercion applies. These are known as local coercions (Luo, 2011). Local coercions make explicit where a coercion can be applied. For instance, if (21) is used to mean (25) or ‘Julie enjoyed writing a book’, we exploit the following two coercions for (21):

(26) coercion BOOK ≪_reading EVENT in (22)
(27) coercion BOOK ≪_writing EVENT in (22)

Note that such interpretations involve different local coercions and can be used in the same context. There is no ambiguity or confusion as to which coercion is to be employed, but we must make clear the scope of each one of the coercions, over what terms they are operative. Like TCL, MTT can use polymorphic or dependent types to get a finer grained account of coercion between non eventuality types and eventuality types. Nevertheless, the uniqueness assumption still applies. In TCL terms, in MTT one can only have one event type e(α, β) for any specification of α and β. TCL allows the type to remain underspecified because it allows for restricted quantificalational types (undescription is treated as an existential quantification over all specifications), whereas MTT does not. More importantly, TCL allows multiple coercions because these are lexically specified and operate on particular predications involving particular occurrences of lexical items, whereas the coercions in MTT are not tied to particular lexical items but are rather part of general world knowledge. The linguistic facts discussed at length in Asher (2011) provide strong evidence that something like TCL’s approach must be right.

Nevertheless, the notion of local coercion in MTT can be used in TCL to account for the puzzling data in (5) and (6). As they stand, both TCL and MTT approaches to coercions make available for subsequent discourse both “coerced content” and the original meaning in a coercion. But we’ve seen that coercions are not all alike. For the container/contents sort of examples in (2), this strategy makes sense: both entities seem indeed to be available for subsequent anaphoric reference. But TCL and MTT make wrong predictions for other coercions like those in (5) or (6) examined in section 2.2; both theories predict that all of those examples should be semantically felicitous when they are not.

The discourse entities that result from coercions are anaphorically available only in certain discourse configuration. To account for the data, one solution explored in Asher and Luo (2012) is to identify two coercion mechanisms. The first is a local coercion that adjusts types only within an argument of the predicate licensing the coercion. As with simple coercive subtyping, this coercion has a very local scope, but its effects are rather different. Using TCL we can make this explicit. As before, we use the TCL mechanisms to introduce a functor on a component of logical form. But while the standard TCL coercion functors makes available both a term of the shifted type as well as a term for the original argument type available for subsequent predications, our new functor does not. It takes an input type a, which is the head type of the original argument DP and an output type b, the shifted type as parameters and has scope over a predication involving a predicate P and an argument x. This functor has the following form:

(28) \( \lambda P \lambda x \lambda \pi C_{a,b}(P(\pi)(x)) \)

The introduction of this functor is triggered by the same type conflicts and type requirements given by a particular predicate in TCL. Applying this functor instead of the one described in 18 to a predication that involves coercions allows type presuppositions in TCL to be justified, because the argument for the predicate is mapped into something of the right type. And so predication succeeds. Using the functor in (28), the logical form for (3), ignoring the type presupposition parameters \( \pi \) for readability, becomes:

(29) \( \exists e \exists x(\text{book}(x) \land C_{\text{book},\text{EVT}}[\text{enjoy}(j, x, e)]) \)

In (29), compared to (19) or its simplified form in (12), we see only a discourse referent for the book, none for the eventuality that is the object argument for \( \text{enjoy} \). This eventuality argument for \( \text{enjoy} \) is not represented by a distinct term in logical form and so, given standard assumptions of dynamic semantics about the treatment of anaphora, is not available as an antecedent for anaphora in the discourse to come.
The operator $C_{\text{book},\text{evt}}$ requires a semantics. Assuming a background dynamic semantics like Dynamic Predicate Logic (Groenendijk and Stokhof, 1991) or compositional Discourse Representation Theory (Muskens, 1996), we also assume a function $c$ associated with our coercion that maps objects of type $a$ into objects of type $b$. In effect, when applied to any variable $x$, $c(x)$ gives us a new term of a different type. Given this, the semantics of our operator is relatively straightforward. Let $\mathcal{M}$ be an intensional model. We can define the update value of a formula of the form $C_{a,b}\phi(t_1, \ldots, t_n)$ as the following relation between world assignment pairs relative to $\mathcal{M}$:

\begin{equation}
\forall \mathcal{M}, \mathcal{W}, f \mid C_{a,b}\phi(t_1, \ldots, t_n) \vdash \mathcal{M}, \mathcal{W}, f \iff \\
\exists g \text{Dom}(g) = \text{Dom}(f) \cup \{c(t)\} \cap \mathcal{M}, \mathcal{W}, f \mid \phi(t_1, \ldots, t_n, c(t)) \mid \mathcal{M}, \mathcal{W}, g
\end{equation}

In dynamic semantic terms such a formula is a test on world assignment pairs; it returns the same world assignment pair if the conditions are satisfied, and otherwise returns nothing. So for instance, the logical form in (29) will be satisfied just in case Julie enjoyed doing something to the book.

To handle coercions of the omelette variety, we have to change coercion operators. Suppose an operator $H_{a,b}$ (for hide) that provides a map for $f$ from type $a$ to type $b$. The data also shows that $H$ must take scope over the whole of the coerced argument (say the entire DP) and the predicate. With the operator $H$, only the result of the map is visible in logical form. The semantics of $H_{a,b}$ is as follows:

\begin{equation}
\forall \mathcal{M}, \mathcal{W}, f \mid H_{a,b}\phi(t_1, \ldots, t_n) \vdash \mathcal{M}, \mathcal{W}, f \iff \\
\exists g \text{Dom}(g) = \text{Dom}(f) \cup \{c^{-1}(t)\} \cap \mathcal{M}, \mathcal{W}, f \mid \phi(t_1, \ldots, t_n, c^{-1}(t)) \mid \mathcal{M}, \mathcal{W}, g
\end{equation}

Our treatment entails that the map from $a$ to $b$ must be bijective. Our theory predicts that there are no omelette type coercions where there is not a one to one correspondence between the objects of the coerced type $b$ and the objects of the original type $a$.

For a sentence like (6a) *The omelette left without paying*, we would have for the first clause (ignoring the anaphoric and presuppositional nature of the definite), which poses some interesting problems in itself:

\begin{equation}
\exists x \text{food, person}[\text{omelette}(x) \land \text{left}(x) \land \neg \text{pay}(x)]
\end{equation}

Note that while the existentially variable is available for subsequent anaphoric reference, its type is that of the result of the coercion—it’s the person.

Besides these local coercion operators, we need a second mechanism, spell out, to account for all the facts. While the lexical and compositional levels introduce local coercion operators when mandated by type conflicts and specific types associated with the predicates as in TCL, spell out takes local coercion operators and fleshes them out into the sort of TCL functors we have seen before with (11) and were introduced in Asher (2011). For coercions like container/contents coercion, call these *robust coercions*, spell out seems to operate without restrictions. For the others considered in this paper, however, discourse constraints constrain when spell out can occur. This leads to two different types of coercions that must be marked in the type system and the typing of predicates: those that always permit spell out and those that only permit spell out under certain constraints.

When certain, “anti-local” discourse configurations obtain, spell out takes these local coercions and replaces them with the more elaborate functors of the standard TCL sort. These functors introduce a new discourse entities of the new types into the discourse context such that they are available for subsequent anaphoric reference, which wasn’t the case with the local functors. Schematically, the transformation is this:

\begin{equation}
C_{a,b}F(t_1, \ldots, t_n, t) \rightarrow \exists x : b(Rxt \land \phi(t_1, \ldots, t_n, x)) \text{ (where } t : a)\text{.}
\end{equation}

We replace our formula prefixed with a local functor with a first order formula, $R$ being specified by the polymorphic type associated with $C$.\textsuperscript{16}

What is the anti-local feature that controls spell-out? The data indicate that the feature has to do with particular discourse configurations, not sentential distance. It also depends on particular coercions that themselves depend on the predicate at issue. We suppose that the basic coercion form with operators like $C$ and $H$ initially introduced in logical form by the type adjustment and the type requirements of particular predicates are transformed via Spell Out into the more familiar material from TCL when an anti-local discourse configuration occurs. For the event based coercions, it is crucial that the discourse unit with the event anaphor relates to the unit with the coercion via a relation that enforces that the

\textsuperscript{16} For our functors of type $H$, a slightly different form of spell out is called for.
eventuality introduced in the coercion is distinct from the eventuality described by the clause with the anaphor. This is guaranteed for example by a relation like Explanation or Result, whose semantics involves a causal relation; if there is a causal relation between \( a \) and \( b \), then \( a \) and \( b \) must be distinct because there are no self-causing facts or events. That is, if there is a causal relation between \( a \) and \( b \), \( a \) and \( b \) must be physical objects, events or have some physical realization—abstract objects do not, according to most philosophers, have causal powers. Furthermore, if a causal relation holds between two physical objects or events must be spatio-temporally distinct, as causal effects take some minimal amount of time, and this spatio-temporal distinctness requires \( a \) and \( b \) to be distinct objects. Contrasts also involve a shift from one fact to another that perforce have different properties.

Relations like Elaboration, where Elaboration\((a, b)\) holds only if \( b \) expands on or gives more details about the fact or event described in \( a \) do not guarantee distinctness, and a relation like Continuation has a semantics according to which \( a \) and \( b \) are both descriptions of the same topical event or fact.

All of the examples in which spell out makes the correct predictions about anaphoric availability with event coercions involve one of these anti-local discourse relations. The examples in which spell out makes the wrong predictions are those that feature relations like Elaboration, Reformulation or Continuation or feature the anaphor and its antecedent within one elementary discourse unit. Consider (5b or (5c) repeated here:

\[
\begin{align*}
(5) & \quad b. \quad \text{Jill started a book. } \text{?The writing } \text{?} \text{That will last for years.} \\
& \quad c. \quad \text{*Jill started War and Peace, which will last for weeks.}
\end{align*}
\]

The second sentence and appositive relative clause in those examples attempt to elaborate on the coerced event. The discourse structure between the two clauses does not have the requisite element of non locality. On the other hand, (5f) involves a relation of Explanation and is predicted to be good. The data in (6) refines the anti-locality hypothesis. Here are the examples again:

\[
\begin{align*}
(6) & \quad a. \quad \text{??The omelette left without paying, although it was very yummy.} \\
& \quad b. \quad \text{??The omelette, which has fresh mushrooms in it, left without paying.} \\
& \quad c. \quad \text{The omelette that had fresh mushrooms in it left without paying; the omelette with ham and cheese paid.} \quad 17 \\
& \quad d. \quad \text{??The delicious omelette has left without paying.} \\
& \quad e. \quad \text{The omelette has ordered. He wants it with mushrooms.} \\
& \quad f. \quad \text{The omelette is getting restless. But you can tell him it’s already ready.} \\
& \quad g. \quad \text{The omelette left without paying because he found it disgusting.} \\
& \quad h. \quad \text{The omelette is enjoying it.} \\
& \quad i. \quad \text{*The omelette ate it with gusto.}
\end{align*}
\]

(6b) also features a relation of Elaboration between the clause with the coercion and the relative clause with the anaphor. In contrast, (6c) features a restrictive relative clause that does not provide a separate discourse constituent. As such, Spell Out cannot be triggered, which is what also happens with (6d,i). Our two step approach to non robust coercions correctly predicts these examples to be infelicitous. It also predicts examples (6f,g) to be good because they feature anti-local discourse structures (Commentary and Explanation). More puzzling, (6e), which has an Elaboration relation between the two clauses, is good. However, this is not an Elaboration of the original meaning of the omelette; it’s an Elaboration of what the omelette qua individual has ordered. This makes all the difference. The anti-locality hypothesis also explains why (6a) is degraded. While (6a) exemplifies a contrast, it is what many researchers on discourse structure would call a Concession (Mann and Thompson, 1987). In SDRT (Asher, 1993; Asher and Lascarides, 2003) concessions are defined in terms of Contrast and Background, and Background requires a topic, which in this case is also the original denotation of the omelette. What is common to all of the bad or local discourse structures is that the element hidden by the local coercion operators is in the discourse topic dominating the discourse constituent in which the anaphor to the hidden element occurs. The bad cases of Elaboration and Background all have the hidden discourse entities in topic position. The hypothesis about when spell out is licensed is: spell out for coercions with non robust coercions occurs only in those discourse structures in which the element hidden by the local coercion operators is not in the topic position (for details on topics in discourse structure, see for instance Asher, 1993; Asher and Lascarides, 2003).

Why should such a constraint on spell out exist? Why should it only exist for some coercions? The container/content coercions are happy in English to provide discourse entities corresponding to the original meaning of the coercing predicate’s argument and discourse entities that are inferred from the type coercion mechanism in the local discourse configurations. Because of its sensitivity to the type of coercion at issue, probably the anti-locality constraint is not a

---

17 Thanks to a reviewer for this example.
general feature of discourse structure. It is rather a feature of particular coercions, particular types of predicates, and probably of particular languages.

A final puzzle involves the mono-clausal contrasting pairs (6h) and (6i). The main structural difference between these examples is that the acceptable (6h) involves an event coercion. The account suggests that this “compound coercion” somehow produces an anti-locality effect and the requisite distance between the real omelette and the term introduced by coercion, the person eating the omelette.

3. Adjective noun combinations

In this last section, I consider a different sort of construction, adjective noun combinations, that resemble verbal coercions but do not always involve coercions in the sense of a repair of a type mismatch. While TCL explains coercions involving adjectives and nouns as in quick cigarette along the lines of verbal coercions, there are shifts in meaning in adjective noun combinations, even when there isn’t a type mismatch.

The vast majority of adjectives in most languages are subsective; i.e., if x is an Adj N, it is an N. Those few adjectives that aren’t subsective, like former prisoner, support inferences that subsective adjectives do under some sort of modal or temporal operator. For example, former prisoners were once prisoners, future candidates are candidates at some future point in time, possible difficulties are difficulties in some epistemic alternative, and fake guns and stone lions appear to be or look like guns and lions.18

On the other hand the head noun can affect the meaning of the modifier, as the following well-known sorts of examples illustrate.

(34) a. flat surface
b. flat country
c. flat tire
d. flat water
e. flat beer

If these data are well-known, the models of analysis for them are controversial. We could assume that adjectives are wildly ambiguous, roughly one sense for each noun with which they can combine. But that would miss certain logical relations. For instance, a mathematically flat surface does have something in common with a flat tire, and even with flat beer. We could say that adjectival types are polymorphic, but that simply labels the problem and doesn’t solve the problem of inferences. There is a sense of flat that might be taken as fundamental, that of a flat surface, which then shifts to a different but related meaning when the adjective combines with nouns of type other than surface.

From these observations, it is roughly clear what the result of a meaning composition for a modifier noun should be. It should look something like this, where M (modifier) is the adjective and N is the noun and assuming that the logical form expressions reflect the types in a natural way:

(35) Adjective Noun combinations: \( \lambda x (O_M(N(x)) \land T_N(M(x))) \)

So to understand this form of meaning combination, we have to specify two sorts of type interactions, the one that gives rise to the functor \( O_M \), and the one that gives rise to \( T_N \).

For subsective adjectives, \( O_M \) is just the identity functor over the type of common noun phrases. How do non-subsective adjectives behave? For temporal non-subsective adjectives like former, future and modal adjectives like possible, likely, it’s pretty clear what \( O_M \) should be. It will be a temporal or modal operator.19 For material modifiers like stone, brass, clay it’s less obvious. These can function subsectively but also non-subsectively; it depends on what the head noun is that they combine with. Brass pots are pots, and stone pots are pots too. But a stone lion can’t be a lion (it’s not a living animal). In fact, it is essentially or necessarily not a lion. This is born out by preliminary experiments over a large corpus using vector models for meanings like those found in Mitchell and Lapata (2008). Bride et al. (2014) calculated a vector for lion en pierre using the matrix approach to adjectives advocated by Baroni and Zamparelli (2010a) over a parsed version of the French FrWak corpus; using the standard cosine distance measure of similarity for vectors, the closest vectors were those of expressions like statue, sculpture, stone, while the vector for lion had a much larger cosine distance from stone lion. However, a stone lion looks much more like a lion than any of the alternatives that lion

---

18 TCL and many other approaches—e.g., Kamp and Partee (1995) and Partee and Borschev (2004)—adopt this view for adjectives and modifiers.

19 Some cases are much less obvious, however. Consider capital vs. cultural capital. Cultural does not seem to be subsective, but it gives some sort of modal frame—within the cultural worlds, X is the capital of the country/world...
suggests—these are associated types the are common hyponyms of Lion’s lowest common super type (Asher, 2011).
Following the analysis of Asher (2011), we get something like this for a generic material modifier Mat:

\[
\lambda P \lambda x \pi \left( P(x) \land \text{made-of}(\text{HD}(\text{MAT}), \text{HD}(P))(x) \right) \land \exists u(\text{Mat}(u) \land \text{made-of}(u, x))
\]

Since the \( \lambda \) abstracted variable \( P \) ranges over first order properties, (36) can combine with the entry for a noun in TCL or with a simple predicate like \( \lambda u \text{lion}(u) \) to yield a \( \lambda \) term for a adjective–noun combination. Asher (2011) argues that in case the head type of the noun modifier construction, \( \text{hd}(N) \) (i.e. the type of the noun), cannot have as a material that given by the modifier, then the predicate has to be understood loosely. Although a stone lion isn’t a real lion, it shares many more features with lions than it does with other animals (shape, typical posture and so on). That is, a stone lion is so called because it is more a lion than the other alternatives evoked by the expression.

Here I take a different approach and code this notion of loose speaking into a particular sort of functor with a well-defined semantics. The reason is that although loose talk is a fact of life, it is difficult to integrate it properly into formal semantic theories. Although the details may get a bit complicated, in principle we can specify \( O_M \) when the types \( \text{MAT} \uparrow \text{HD}(P) = \perp \). The type combination specifies the operator \( O_M \) to something like it appears that or resembles as in (37), since loose talk typically makes use of superficial properties that serve to infer that something of a particular type. I offer one way to spell this out in (38) for such a modification of lion, where the object should share most of the superficial properties (the \( X \) properties) with real lions. We could further restrict \( X \) to explicitly compare the properties that our target shares with respect to lions compared to the properties it shares with the other alternatives:

\[
O_{\text{made-of}(\text{stone,lion})} := \lambda P \lambda x \text{looks-like}(P(x))
\]

\[
O_{\text{made-of}(\text{stone,lion})} := \lambda P \lambda x \text{MostQ}(X(Q)) \rightarrow (\text{Gen } z(P(z), Q(z)) \rightarrow Q(x))
\]

When applied to a common noun phrase denotation as in the schema (35), this functor takes \( \lambda x \text{lion}(x) \) and transforms it into the property of things that have most of the superficial properties that lions typically or normally have. This is in effect a kind of coercion, though not one that would fit under traditional analyses of coercion.20 The restriction on properties \( X \) makes the property quantification range over those superficial properties that lions or the alternatives have. For material modifiers, on the other hand, the functor \( \tau_N \) from schema (35) is just the identity functor: stone lions really are stone, or made of stone, as are stone jars, stone houses, etc. This follows from theses about essential materialism as detailed in Asher (2011). Putting these observations together, we now have the form for the predication involving a material modifier and a noun like stone lion using the schema (35) and our specifications for \( O_M \) in (37) and \( \tau_N \) as the identity functor:

\[
\lambda x \text{MostQ}(X(Q)) \rightarrow ((\text{Gen } z(\text{lion}(z), Q(z)) \rightarrow Q(x)) \land \text{stone}(x))
\]

Our formalism allows us to introduce non trivial functors for both \( O_M \) and for \( \tau_N \). However, this seems to happen very rarely if ever. One possible case involves evaluative modifications like bad mathematician. If one’s a really bad mathematician, maybe one isn’t a mathematician at all. But one need not be bad tout court either, if one is a bad mathematician. But at least one is really bad with regards to being a mathematician. However, I have found it hard to find such examples, and so the following probably familiar hypothesis holds by and large: in a modifier noun combination, either the modifier or the noun literally holds of the object—otherwise the modifier noun combination begins to lose descriptive content and functions more like a proper name (The Holy Roman Empire comes to mind, which was neither holy, Roman nor an empire). That is, one of either \( O_M \) or \( \tau_N \) must be trivial. Why, one might ask, is that? One hypothesis, unavailable to the proponents of loose talk, is that an assertion comprising a modifier noun construction, which is, on the present view, engaged in conveying at the least a literal content, must be anchored somewhere in plain description; if the noun is modified by the adjective in the way that we have seen for material modifiers, then the literal modifier meaning is used to drive the introduction of the non trivial functor value for \( O_M \). Conversely, a subsecutive adjective must select a subset of the denotation of the head noun \( N \). This subset should consist of elements that have the property associated with \( M \) to a greater degree than other elements in the denotation outside that subset in a way that is pertinent for elements of \( N \). In either case, the meaning of either the modifier or the noun must be literally apply to the objects in the set picked out by the common noun phrase. While one could imagine a procedure in which there is mutual interference and a resulting fixed point equation for the meaning of a modifier noun combination, that doesn’t seem to happen. Clearly this operation is much more complex than the simple adjustments to types and logical form that ensue when one of the expressions in a noun modifier combination is taken literally. Perhaps the fixed point operation is too complex for natural language processing and too prone to misinterpretation by fallible agents for reliable communication.

\[\text{Please cite this article in press as: Asher, N., Types, meanings and coercions in lexical semantics. Lingua (2015), }\]

http://dx.doi.org/10.1016/j.lingua.2015.01.001

\[\text{Partee (2010) also advocates something like this view, though the details are quite different from those of the approach developed here.}\]
With this in mind, let’s return to the question of the determination of the functor $T_N$ for subsective adjectives, where the functor $O_M$ is the identity functor. Let’s consider once again the subsective adjective like flat. A flat country is closer to a mathematically flat surface than one that is not, at the scale relevant for measuring the surface of countries. A flat tire has a portion that is closer to a flat surface than any portion of a normally inflated tire. Flat water and flat beer have still unperturbed surfaces that are flat by comparison to bubbly ones. In each one of these cases, the way the adjective is applied changes slightly in virtue of the type of the head noun, but there is also always a common reference to a standard mathematical notion of flatness (0 curvature, or close to 2 dimensionality). Unlike the cases of coercion we have looked at before or the cases of the “loose-talk” material modifications, the change in the meaning of flat depending on what type of object it applies to doesn’t arise from a type conflict: flat applies to physical objects; and tires, countries and beer are all physical objects.

Adjective noun compositions involve an operation similar to coercion; but since there is no type conflict, $T_N$ is an operator that always takes the head type of the modified noun as a parameter and provides on that basis an adjustment to logical form. Given the form of a TCL entry for a common noun, the type information from the noun is passed to the modifier, so implementing such an operator in TCL is straightforward. Here is the entry for a noun $N$ whose type presuppositions on its head variable is $\lambda$ and where $P$ ranges over adjectival denotations (simple predicates) and $s$ is the type presupposition introduced by the adjective $P$:

$$
\lambda x \lambda \pi \lambda \tau \lambda \pi (T_N(P(x, \pi \times P : APPLIES-TO(HD(N)))) \times x : A) \land N(x, \pi \times x : B))
$$

In particular, let’s look at the entry for beer:

$$
\lambda x \lambda \pi \lambda \tau \lambda \pi (P(x, \pi \times P : APPLIES-TO(beer) \times x : LIQUID \sqcap MASS) \land N(x, \pi \times B))
$$

Now let’s look at the entry for flat, whose type presupposition is just that the object it applies must have at least three dimensions:

$$
\lambda x \lambda \pi \lambda \tau \lambda \pi (flat(x, \pi \times x : 3-D))
$$

Combining the $\lambda$ terms in (42) and (41) yields:

$$
\lambda x \lambda \pi \lambda \tau \lambda \pi (flat(x, \pi \times flat : APPLIES-TO(beer) \times x : LIQUID \sqcap MASS) \land N(x, \pi \times x : 3-D))
$$

I suppose now that the functor $T_N$ is introduced in virtue of the fact that the type flat must combine with the type beer. Such a functor does not result from a type mismatch but rather from adjustments resulting from the fine-grained types associated with adjectival and noun. This functor results from the passing of the most specific type associated with the head noun to the adjective—in TCL type information is passed from the head to its arguments and so this fits with TCL’s general architecture. I now show the result of the triggering of the relevant functor in the case of flat beer. The procedure is general, though the particular spell-out rules depend on course on the specific type of the head noun.

$$
\lambda x \lambda \pi \lambda \tau \lambda \pi (T_{beer}(flat(x, \pi \times LIQUID \sqcap MASS) \land beer(x, \pi \times P))
$$

(45) spelling out the functor $T_{beer}$:

$$
\forall z, w ((portion-of-beer(z, \pi \times x : 3-D) \land surface(w, z, \pi \times x : 3-D) \land has-bubbles(z, \pi \times x : 3-D) \rightarrow flatter(y, w)) \land beer(x, \pi \times x : 3-D))
$$

This approach predicts that we should see an influence either of the adjective on the noun’s denotation when we have a non-subsective modifier, or an influence of the noun on the modifier’s meaning for subsective adjectives. More generally, the approach predicts an often non-trivial shift in the meaning of a modifier as it combines with various nouns or vice versa. This coincides with findings in distributional semantics for adjective noun combinations following Baroni and Zamparelli (2010a), using cosine similarity measurements, as well as with anecdotal observations.

A puzzle from the perspective of coercions is why such type adjustments take place in modifier noun combinations. In the cases of coercion we have considered (but this seems to hold for all examples of coercions), the introduction of functors comes about because of a type conflict between the selectional restrictions of the predicate and the type of its argument. But this is not the case in general for modifier noun combinations where we see an influence on the adjective’s denotation or the noun’s, except in the case of material modification where the matter assumed to the objects denoted by the head noun is essentially incompatible with the type of those objects. So how are these shifts to the modifier meaning
triggered? It appears that modifiers have a contextually sensitive meaning that depends on the fine-grained type information of what they modify. On the one hand, this confirms the TCL view that it is the head noun that passes its type information to the adjective, but on the other it shows that we cannot just understand meaning shifts as coercions forced by incompatibilities between the type presuppositions imposed on an argument by a predicate and the type presuppositions of the argument itself. Semantic composition allows for the introduction of new meaning even without shifts in type that are relevant for the selectional restrictions. Furthermore, this is not an aspect of modification that is restricted to a few adjectives. Large coverage distributional approaches show that in adjective noun combinations, the sense of the adjective relevant for the selectional restrictions. Furthermore, this is not an aspect of modification that is restricted to a few adjectives. Large coverage distributional approaches show that in adjective noun combinations, the sense of the adjective

4. Conclusions and prospects

In this paper I've examined two sorts of meaning composition, mostly from the perspective of the Type Composition Logic, though I have also compared this framework with recent developments in Modern Type Theory at least with respect to verbal coercions. TCL's translation of type interactions into standard logical forms with a denotational semantics offers us a means for investigating fine-grained shifts in the meanings of modifiers and their targets.

The common core of the analysis in the two approaches is that type interactions give rise to functors that may repair a type mismatch in the case of coercions or adjust the meaning of the modifier or the noun in ways that still license certain important inferences that depend on the modifier or noun's (more generally the predicate or the argument's) core meaning. Experiments involving distributional semantics show that such meaning shifts for adjective noun combinations are pretty widespread, though the measure of similarity used there cannot say exactly what inferences are preserved or why. To date, distributional methods can only provide us with rough comparisons, but they do show that there are shifts and that these shifts are reflected in usage across very large corpora. If we follow the results from distributional models, we should predict other shifts in meaning, say between verb and argument combinations, even in cases where there is no type conflict. The approach I have developed for adjective noun combinations should translate to that type of predication as well: fine-grained type information will determine functors perhaps both for arguments and predicates. This leads to a more general point. I believe that by combining formal and distributional methods, we can develop robust but explanatory models of meaning combination that have wide coverage. But that is a subject for another paper.

Acknowledgements

This paper was supported by the ANR grant Polynmie. Thanks also to Zhaohui Luo for many comments and for the examples in Chinese and for his work on MTT theories. Thanks also to Stergos Afantenos for some nice examples.

References


Please cite this article in press as: Asher, N., Types, meanings and coercions in lexical semantics. Lingua (2015), http://dx.doi.org/10.1016/j.lingua.2015.01.001