Running head: Metaphors, Logic and Type Theory

Metaphors, Logic and Type Theory

Josef van Genabith

Dublin City University

Computer Applications

Dublin 9, Ireland

josef@compapp.dcu.ie

Abstract

Metaphorical use of language is often thought to be at odds with compositional, truth-conditional approaches to semantics: after all, most metaphors are literally false. In this paper we sketch an approach to metaphors based on standard type theory. Our approach is classical: we do not invent a new logic. The approach models sense extension in a simple and elegant way: the properties (supertypes) shared between tenor and vehicle include the extensions of at least both. The original predicates remain unchanged. Our approach captures an asymmetry between metaphor and simile: the literal interpretation of a metaphor comes out as (mostly) false while its non-literal interpretation is that of a corresponding reduced simile. A compositional syntax–semantics interface is provided and a deductive account of metaphor resolution is outlined. The approach readily translates into a simple computational implementation in Prolog. We discuss how our approach addresses issues of generalisation, feature selection, asymmetry, tension, trivialisation, prototypicality, truth conditions, comprehension and generativeness.

Metaphors, Logic and Type Theory

Non-literal use of language such as metaphor is usually thought to sit uneasily with formal, truth-conditional semantics in the Montagovian tradition (Montague, 1973). Most metaphors are simply literally false.[1] Consider e.g. the following established metaphor, its formalisation in First-Order Predicate Logic (in FOPL quantification is restricted to range over individuals) and associated truth conditions:

(1) ‘‘`John is a fox.`’’ | $fox(j)$ | $[\![fox(j)]\!] = 1$ iff $[\![j]\!] \in [\![fox]\!]$

The formula $fox(j)$ can be glossed as: the one-place predicate fox (the FOPL translation of `fox`) is predicated of the logical constant j (the FOPL translation of `John`). Equivalently, the formula states that j has the property fox. Formulas in FOPL are interpreted in models. A model is a set theoretic construct consisting of a universe of interpretation (a set of objects; also referred to as the domain) and an interpretation function which specifies which constants are interpreted as which objects in the universe and which predicates are interpreted as which subsets (of individuals or n-tuples, depending on the number n of arguments particular predicates take) in the universe. The interpretation of a constant or predicate symbol is also variously referred to as the denotation or extension of the constant or the predicate symbol. A model fixes the interpretation of basic constituent expressions (the vocabulary, if you like). Complex expressions, i.e. formulas, are interpreted in terms of a recursively specified function (often represented as $[\![\cdot]\!]$) which follows the syntactic formation rules of FOPL. The base cases of this function are provided by the interpretation of constants and predicate symbols given by the model.

On this account the interpretation of (1) is true if and only if the denotation $[\![j]\!]$ of the logical constant j (the translation of `John`) is an element of the denotation $[\![fox]\!]$ of

the one-place predicate fox (the translation of `fox`). Put differently, (1) is true if and only if $\{[\![j]\!]\} \cap [\![fox]\!] \neq \emptyset$.

This, however, is not the case that obtains in the literal reading of (1) involving, as it does, a predication of a property to an individual not in the extension of the property predicated (to be fully explicit: here we are, of course, assuming that John is human). Several responses are possible. For all their differences, most approaches to metaphor assume that metaphor invites the determination of a similarity or likeness between tenor and vehicle. One line of thought maintains that metaphor is a comparison statement (Aristotle, 1952) that can be analysed as a reduced or elliptical simile, e.g. (Fogelin, 1988). On these accounts (1) corresponds to (2) paraphrased in (3), or, following Black's "system of associated commonplaces" (Black, 1962), to (4), paraphrased in (5):

(2) `John is like a fox.`

(3) `John has some of the properties of foxes.`

(4) `John is like a typical fox.`

(5) `John has some of the typical properties of foxes.`

Paraphrases (3) and (5) are readily translatable into standard type theory (Church, 1940) and a compositional syntax-semantics interface can be set up. This will allow us to parse natural language strings automatically into literal and metaphorical meaning representations and this is one of the themes developed in the present paper. Standard type theory is a higher-order logic (HOL) based on the typed λ-calculus. HOL (rather than FOPL) is required because paraphrases (3) and (5) quantify over properties `...some of the properties ...` (i.e. sets) rather than just individuals. Versions of HOL have been the standard choice of representation formalism in much formal semantics in the Montagovian tradition.

Interpretation of metaphor as corresponding reduced simile has been objected to on a number of grounds. We discuss how our approach addresses issues of generalisation, feature selection, asymmetry, tension, trivialisation, prototypicality, truth conditions, comprehension and generativeness.

Type-Theory $\mathcal{TT}$

The type theory $\mathcal{TT}$ we employ is little more than a sugared version of the typed λ-calculus (see e.g. (Church, 1940), (Gamut, 1991b)). The basic idea in type theory is that based on a set of primitive types (in the simplest version a type e of entities – or individuals – and a type t of truth values) logical connectives, predicates, arguments and quantifiers are represented in terms of functions over those basic types. n-place relations, e.g., can easily be coded as $n+1$-place functions. The typing regime is designed to avoid paradoxes and inconsistencies which could otherwise arise due to the considerable expressive power of HOL. Below we briefly sketch simple extensional type theory which is going to provide our representation formalism. The set of types $\mathcal{T}$ is defined as $e, t \in \mathcal{T}$ and if $a, b \in \mathcal{T}$ then $\langle a, b\rangle \in \mathcal{T}$ (this is the type of functions from type a objects to type b objects). The basic vocabulary of $\mathcal{TT}$ has sets of variables Var_τ and constants Con_τ, for each $\tau \in \mathcal{T}$. The syntax closes $\mathcal{TT}$ under application, abstraction, the logical connectives and quantification. Interpretation is relative to models $\mathcal{M} = \langle \mathcal{D}, \Im\rangle$ where $\mathcal{D}$ is a domain of individuals and $\Im$ an interpretation function interpreting constant symbols. Types are interpreted as function spaces (domains). Interpretation domains $\mathcal{D}_\tau$ for types τ are defined as $\mathcal{D}_e := \mathcal{D}$, $\mathcal{D}_t := \{0, 1\}$ and $\mathcal{D}_{\langle a,b\rangle} := \mathcal{D}_b^{\mathcal{D}_a}$. Given a model $\mathcal{M} = \langle \mathcal{D}, \Im\rangle$ with $\Im : Con_\tau \to \mathcal{D}_\tau$ and $g : Var_\tau \to \mathcal{D}_\tau$ (for each type τ) the interpretation function $[\![\cdot]\!]$ is defined as follows:[2]

1. $[\![c_a]\!]^{M,g} = \Im(c_a)$; $[\![x_a]\!]^{M,g} = g(x_a)$
2. $[\![\varphi_{\langle a,b\rangle}(\psi_a)]\!]^{M,g} = [\![\varphi_{\langle a,b\rangle}]\!]^{M,g}([\![\psi_a]\!]^{M,g})$

3. $[\![\lambda x_a \varphi_b]\!]^{M,g}$ is that function h such that for all $u \in \mathcal{D}_a$, $h(u) = [\![\varphi_b]\!]^{M,g[x/u]}$
4. $[\![\neg\varphi_t]\!]^{M,g} = 1$ iff $[\![\varphi_t]\!]^{M,g} = 0$
5. $[\![(\varphi_t \wedge \psi_t)]\!]^{M,g} = 1$ iff $[\![\varphi_t]\!]^{M,g} = 1$ and $[\![\psi_t]\!]^{M,g} = 1$
6. $[\![\forall x_a \varphi_t]\!]^{M,g} = 1$ iff for all $u \in \mathcal{D}_a$ $[\![\varphi_t]\!]^{M,g[x/u]} = 1$

Axiomatisations of $\mathcal{TT}$ are incomplete under interpretation in standard models (admitting the full function spaces). Sound and complete axiomatisations of $\mathcal{TT}$ are provided for general models (Henkin, 1950). For readability, we will often suppress type annotations in the formulas below.

Expressing Similes in $\mathcal{TT}$

On the most natural reading of the simile interpretation (2) of (1) the object NP is given a generic (`all / most / typical /` bare plural) interpretation:

(6) `John has a property which is a property of (all / most / typical) foxes.`

For expository purposes and reasons of space, below we approximate the genericity of the object NP argument by simple universal quantification. More sophisticated (and appropriate) treatments are possible, see for example (Carlson and Pelletier, 1995), and in a later section we outline an interpretation based on a prototype, i.e. a cultural stereotype, analysis. With this proviso (6) is approximated by the following $\mathcal{TT}$ expression:

(7) $\exists P(P\ j \wedge \forall x(fox\ x \rightarrow P\ x))$

This $\mathcal{TT}$ formula can be glossed as follows: there exists a property P which holds of j and P is a property of all foxes. (7) comes out true if there exists a property P (simple or complex) denoting a subset of the domain of entities which includes both the extension of j and the (members of the) extension of the fox predicate:

(8) $[\![\exists P(P\ j \wedge \forall x(fox\ x \rightarrow P\ x))]\!] = 1$ iff there exists a P such that $[\![fox]\!] \cup \{[\![j]\!]\} \subseteq [\![P]\!]$

Sense Extension, Supertypes, Generalisation and Feature Selection

Our analysis captures sense extension in a simple and elegant way. The extension of P is a set that minimally includes both the extension of j and the elements in the extension of fox. Notice, however, that the extension of the original fox predicate itself remains unchanged. The property P is what extends fox and additionally includes at least the extension of j. P is a supertype of fox and the minimal type that includes j. In other words, P generalises fox and the minimal type that includes j.

If instead we had opted for a non-classical approach and extended the denotation of the fox property itself to include that of j we would be faced with the following problem: Assume that all foxes have bushy tails. If the extension of fox were to include that of j we could prove that John has a bushy tail, clearly an undesirable result if, as we are assuming in our metaphor scenario, John is decidedly a member of homo sapiens. What is worse, if our axiomatisation of background knowledge includes a statement to the effect that John is human as well as a statement that the categories human and fox are disjoint, then extending the fox predicate to include j leads to inconsistency. Notice that given the same scenario in our approach such inferences do not go through. (7) constrains the shared property P to hold of both the (original) set of foxes and (the disjoint singleton set of) John. Assuming that John is human, the joint property P cannot be instantiated to that of having a bushy tail. If it was, it would falsify the conjunction in (7). Similarly, inconsistency of the form described above cannot arise because our approach does not extend the fox predicate.

Notice further that our analysis naturally captures a feature selection process often attributed to metaphor, most famously perhaps in Black's analogy (Black, 1962) between metaphor interpretation and looking at the stars through an etched piece of smoked glass. Whatever the property variable P is instantiated to, formula (7) minimally requires that it generalises the fox property and the properties of John. That is, the property abstracts

away from what is idiosyncratic to the fox property and j to find properties that are common to both. This is, of course, related to the point raised above and the reason why properties which are not shared (such as having a bushy tail) are suppressed. Feature selection theories have been refined to include graded salience mechanisms (e.g. (Ortony, 1979), (Thomas and Mareschal, 1999)). Such can be addressed by extending our aproach to Probability Logics (e.g. (Adams, 1998)).

You cannot See what is not there ...,Truth Conditions and Asymmetry

On the other hand, our analysis requires that P can only be instantiated to shared properties that are already there. To use Max Black's analogy once again, in this approach the smoked glass (and its clear lines) will not allow you to see things that are not there in the first place. You might not have been aware of them but they have been there all along. It is important to notice that first and foremost the analysis developed in the present paper provides a truth conditional account of metaphorical meaning analysed as reduced simile. It does not provide an account of an agent processing a metaphor. Logic can, of course, be used to extend it to one: intuitionistic, constructive, modal and dynamic logics provide natural settings for modelling information growth and update (e.g. (Jaspars, 1994), (Vogel, 2000)). For our present purposes we follow a more confined programme: in a later section we provide a deductive account of metaphor resolution (i.e. instantiation of P relative to an existing axiomatisation of background knowledge).

On what is not obviously but on closer inspection the same topic, it has often been observed that metaphors are asymmetric (Ortony, 1979): `lawyers are sharks` is not the same as `sharks are lawyers`. By contrast, our approach is symmetric: again, this is because the account developed here provides truth conditions and not a model of the dynamics of an agent's knowledge states under metaphor comprehension.

Tension, Trivialisation, Minimal Extension and Prototypicality

Tension is a characteristic quality attributed to metaphor (e.g. (Davidson, 1984)). Tension derives from the fact that (i) most metaphors are literally false, (ii) literal meaning is still active in non-literal interpretation and (iii) metaphors have an open-ended quality, i.e. precisely which meaning is intended is uncertain. These aspects feature in the analysis offered here: the literal meaning of (1) is $fox(j)$, literal meaning components (fox, j) feature prominently in the representation of the non-literal meaning of (1) in formula (7) and the shared property P is existentially quantified, i.e. we know there should be some property which is shared by tenor and vehicle but we don't know exactly which.

Open-endedness of interpretation, one of the characteristic qualities of metaphor, does not extend to trivial likeness. In fact, trivial likeness has been fielded against analysing metaphor as elliptical simile (Davidson, 1984): "... everything is like everything and in endless ways." While I disagree with Davidson, whose objection relies on (i) the implication that if similarity was trivial then all similarity statements would be trivial and (ii) the false premise that similarity is trivial (the second premise is contradicted by the fact that in most communicative situations where agents use similarity statements the intended and communicated similarity is entirely non-trivial – in other words, similarity is a useful concept), triviality does indeed strike at the formal level: notice that the domain of interpretation (the set of entities) is a set which trivially includes the extension of j and the extension of the fox predicate. From this it follows that a universal property such as $\lambda x.x = x$ (the property of being identical to oneself) trivialy satisfies (7). While it is arguable that trivialisation is the limit case of non-literal use of language, trivialisation of this kind can be ruled out by strengthening the translation to require that P not be instantiated to a universal property, e.g.:

(9) $\exists P(P\ j \wedge \forall x(fox\ x \rightarrow P\ x) \wedge \neg \forall y P\ y)$.

While this move rules out the most trivial (i.e. the universal) properties and ensures that (9) is contingent, it still admits of possibly infinitely many other shared, potentially trivial properties such as e.g. the property of not being identical to my fridge[3] (or indeed any entity described in a background knowledge axiomatisation other than John or any of the foxes). Notice, however, that such inferences crucially depend on a $\kappa^i \neq \kappa^j$ for $i \neq j$ (where κ a metavariable over constant symbols of type e) axiom schema. The schema is optional and requires that distinct constant symbols are interpreted as distinct entities. If we want to rule out a possible interpretation of (1) as `John is similar to foxes in that they are all not the same as my fridge` (which in some bizarre context might in fact be the desired interpretation) we need to switch off (i.e. ignore) the constant axiom schema (if present). Formally this corresponds to structure mapping approaches to metaphor (Falkenheiner et.al., 1989), (Veale and Keane, 1992) not, or only selectively, or only implicitly encoding inequality statements of the sort at stake. Everything else being equal, the type theory based approach developed here and the structure mapping based approaches are generative. They will produce as many interpretations as are admitted by their background knowledge axiomatisations or (in the case of the mapping approaches) knowledge graphs. Generative capacity can be curtailed or extended by axioms or restrictions on proof depth (both options are in fact availed of by mapping approaches in the form of selective knowledge graph coding and limits on recursive computations/graph matches). In addition, in the type theory approach we can curtail generative capacity by strengthening the translation, as in (9). As a further example, consider how a translation can enforce a notion of minimal extension:

(10) $\exists P(P\ j \wedge \forall x(fox\ x \rightarrow P\ x) \wedge \forall Q((Q\ j \wedge \forall x(fox\ x \rightarrow Q\ x)) \rightarrow (P\ j \rightarrow Q\ j)))$.

This translation of (1) requires that the joint property P shared between tenor and vehicle is minimal in the sense that it implies all other shared properties Q.

Before moving on to prototypicality, notice that in contrast to some other (feature based) approaches (e.g. (Thomas and Mareschal, 1999)) our type theory approach does not distinguish between simple and complex properties (in type theory complex properties model relations and relational structure, e.g. (14)). Indeed, from the type theory perspective such a distinction is somewhat artificial. In our approach the properties generated are those that can be proved from whatever is axiomatised. These include simple and complex ones. It is here (in the complex properties) that recursive sub-metaphors can get involved in an interpretation.

In our translations so far we have assumed that the vehicle contributes a generic or a typical property (and in fact we have glossed over the difference between the generic and the typical and, for expository purposes approximated both in terms of universal quantification). It has been observed (e.g. (Black, 1962)) that often what is at stake in metaphor interpretation are cultural stereotypes taking the form of stereotypical individuals or prototypes (rather than definitions of classes in terms of necessary and sufficient conditions). On this account, (1) is likely to be interpreted as stating that `John is clever` and this interpretation derives from comparing John to a prototype FOX. In the words of one of the anonymous reviewers: "The metaphor compares John to an archetype of fox, a cultural model that owes as much to Aesop as to Darwin." This intuition can be integrated into the type theoretic approach. What is required is an axiomatisation of the cultural stereotype FOX. To do this with any degree of confidence requires a psycholinguistic or cognitive theory of cultural stereotypes/prototypes which is beyond the more confined concerns of the present paper. Give such an axiomatisation in the form of e.g. $prty\ fox\ P$ statements where not surprisingly $prty$ (short for prototype) is of the type of a generalised quantifier (Barwise and Cooper, 1981) $(\langle\langle e,t\rangle,\langle\langle e,t\rangle,t\rangle\rangle$ pairing a property – i.e. a class, here fox^4 – with what are its perceived prototypical properties P), the metaphorical meaning of (1) is captured by:

(11) $\exists P(P\ j\ \wedge\ prty\ fox\ P)$

This translation guarantees that the shared property derives from the axiomatisation of the prototypical concept FOX, which is often what is encoded in the knowledge graphs in structure mapping approaches.

In the next section we show how our analysis generalises from simple copula constructions to more complex predications.

Complex Predications

The formulae in (7), (9) and (10) encode a simple supertype/sense extension analysis of metaphors involving predicative uses of the copula `be`. As pointed out, any instantiation of the unary predicate P that makes (7), (9) and (10) true denotes a superset including both the denotation of j and the elements in the denotation of fox. It is here that the sense extension dimension of metaphor is located in our approach. The basic idea can easily be generalised to cover more complex predications as exemplified by the well-worn

(12) ``My car drinks gasoline.''

To a first approximation and following the lead of the approach developed above the non-literal use of (12) can be paraphrased as

(13) `My car and gasoline stand in a relation which is a property of all drink relations.`

The relation in question is probably something like the *consume* relation. Every *drink* event is also a *consume* event (but not vice versa). (13) is readily formalisable. Here we translate the definite possessive NP `my car` as the constant c and simplify the mereological NP `gasoline` as g:[5]

(14) $\exists R\ (R\ g\ c\ \wedge \forall x \forall y(drink\ y\ x \rightarrow R\ y\ x))$

R is of type $\langle e, \langle e, t\rangle\rangle$, i.e. it is a binary relation between entities. As was the case with the simple predication in (7) above, (14) is trivialised by the universal relation $\Re$ (where e.g. x is related to y if x is identical with itself and y is identical with itself). Following (9) this can be ruled out as follows:

(15) $\exists R\ (R\ g\ c\ \wedge \forall x \forall y(drink\ y\ x \rightarrow R\ y\ x) \wedge \neg \forall x \forall y\ R\ y\ x)$.

Following the approach developed in the previous section, the translation can be strengthened to requiring minimal or proto-typical instances of 2-place relations R relative to *drink*.

The *consume* relation provides one of the instantiations of R in (14). Notice that (14) fixes a potential selection restriction violation between *drink* and its subject NP (- *animate*). Assume that *drink* subcategorises for a (+ *animate*) subject NP. (14) forces R to generalise *drink* so that it can apply to `my car` (- *animate*) and `gasoline`. Further, by itself (14) does not support any inference as to excessive amounts of consumption often attributed to (12). Example (12) is similar to the following which was suggested by one of the anonymous referees as a challenge for the approach:

(16) ``I wrestled with the idea.''

Appendix B provides an extension of the Prolog implementation of the compositional syntax – semantics interface presented below which treats example (16) analogous to (12):

(17) `Myself and the idea stand in a relation which is a property of all wrestling relations.`

Resolution

The reduced simile reading $\exists P(P\ j \wedge \forall x(fox\ x \rightarrow P\ x))$ of (1) is weak and trivialised by the universal property. Trivialisation can be excluded in a number of ways as exemplified

in (9), (10) and (11). Trivial use of simile (and metaphor in the reduced simile account) in actual communicative situations is probably quite rare.[6] What makes simile and metaphor interesting is the task of finding non-trivial (i.e. informative) instances of the property P shared between tenor and vehicle. From the existentially quantified formula offered as a reduced simile reading of (1) we cannot deduce much: existential quantification over P amounts to a (possibly infinite) disjunction over suitable predicates of the type of P whose extension is required to include both tenor and vehicle. However, rather than deriving inferences from the reduced simile reading, we can look for proofs that given some background theory (premises in a knowledge base) allow us to deduce the reduced simile reading. Such proofs contain candidate instances of shared properties that enable us to existentially quantify over them. Consider the following simple example (we use the universal quantification approximation of genericity):

$$clever\ j,\ \forall x(fox\ x \rightarrow clever\ x) \vdash \exists P(P\ j \wedge \forall x(fox\ x \rightarrow P\ x))$$

In order to find suitable resolvents $[P = clever]$ we have to inspect proofs. The question is now is there a systematic (i.e. automatic) way of searching for and inspecting such proofs? A signed tableaux proof of the above inference looks as follows:

1	T	$clever\ j$				
2	T	$\forall x(fox\ x \rightarrow clever\ x)$				
3	F	$\exists P[P\ j \wedge \forall x(fox\ x \rightarrow P\ x)]$				
4	F	$clever\ j \wedge \forall x(fox\ x \rightarrow clever\ x)$				
5	F	$clever\ j$	|	6	F	$\forall x(fox\ x \rightarrow clever\ x)$

The trick here is, of course, in the step from line 3 to line 4 in the tableaux. We know that in order to close the tableaux we need to find formulas corresponding to lines 1 and 2 but signed F. However, ideally, we do not want to rely on human intelligence and insight to guide and inspect proofs. This is where free variable tableaux come to the rescue.

Without going into great detail (Fitting, 1996) the basic idea is to delay instantiation of especially introduced variables as long as possible in the development of a tableaux, ideally until closure of a branch. Tracking such variables provides candidate resolutions. A free variable tableaux version of our proof is given below (the predicate variable introduced in going from step 3 to 4 is Π):

1	T	$clever\ j$				
2	T	$\forall x(fox\ x \rightarrow clever\ x)$				
3	F	$\exists P[P\ j \wedge \forall x(fox\ x \rightarrow P\ x)]$				
4	F	$\Pi\ j \wedge \forall x(fox\ x \rightarrow \Pi\ x)$				
5	F	$\Pi\ j$	|	6	F	$\forall x(fox\ x \rightarrow \Pi\ x)$

This tableaux can be closed by matching lines 5 and 1, and lines 6 and 2, thereby instantiating Π to *clever*, which yields a candidate resolution of P.

Notice, that there is a striking parallel between our deductive approach and structure mapping ($\mathcal{SM}$) approaches such as (Falkenheiner et.al., 1989), (Veale and Keane, 1992), summarised as:

$\mathcal{LOGIC}$:	Premises	$\vdash$	Reduced Simile
$\mathcal{SM}$:	Knowledge Base Graph	$\supset$	Metaphor Graph

where $\supset$ is subgraph isomorphism. What differentiates the two approaches is that structure mapping approaches usually intend to give an account of the dynamics of metaphor comprehension whereas our approach explicates truth conditions. As pointed out, logic (intuitionistic, modal or dynamic) can be used to model the dynamics of comprehension but this is beyond the more narrow confines of the present paper.

A Compositional Syntax—Semantics Interface:

In this section we show that the different readings (both literal and metaphorical) associated with (1) and (12) do not come out of thin air but can be computed in a

systematic fashion given a syntactic analysis of the strings at stake. A compositional syntax-semantics interface is specified by a pairing of syntactic formation and semantic translation rules and a specification of the translation of lexical elements. The translation function is indicated $^\circ$:

$$\begin{array}{lcllcl} S & \rightarrow & NP\ VP & S^\circ & := & NP^\circ(VP^\circ) \\ VP & \rightarrow & V\ NP & VP^\circ & := & V^\circ(NP^\circ) \end{array}$$

We assume a generalised quantifier (Barwise and Cooper, 1981) type analysis of NPs.

$$NP \rightarrow \texttt{john}, \texttt{gasoline}, \texttt{my car}, \texttt{a fox} \qquad V \rightarrow \texttt{is}, \texttt{drinks}$$

The type theory translations of the lexical symbols of the grammar are:

$$\begin{array}{lcllcl} \texttt{john}^\circ & := & \lambda P.P\ j & \texttt{gasoline}^\circ & := & \lambda P.P\ g \\ \texttt{my car}^\circ & := & \lambda P.P\ c & \texttt{a fox}^\circ & := & \lambda P.\exists x(fox\ x \wedge P\ x) \\ \texttt{a fox}^\circ_{gen} & := & \lambda P.\forall x(fox\ x \rightarrow P\ x) & \texttt{a fox}^\circ_{\pi} & := & \lambda P.(pty\ fox\ P) \\ \texttt{is}^\circ & := & \lambda P \lambda x P \lambda y(x = y) & \texttt{is}^\circ_{\mu} & := & \lambda Q \lambda z \exists P(P\ z \wedge Q\ P) \end{array}$$

$$\begin{array}{lcl} \texttt{is}^\circ_{\mu,\neg tr} & := & \lambda Q \lambda z \exists P(P\ z \wedge Q\ P \wedge \neg \forall x P\ x) \\ \texttt{drinks}^\circ & := & \lambda Q \lambda x Q \lambda y\ drink\ y\ x \\ \texttt{drinks}^\circ_{\mu} & := & \lambda Q \lambda x Q \lambda y\ \exists R(R\ y\ x \wedge \forall z \forall w(drink\ z\ w \rightarrow R\ z\ w)) \end{array}$$

In this grammar we have glossed over the internal complexity of NPs. We assume that an indefinite NP such as `a fox` is ambiguous between an existential, a universal (gen – our simplified, quasi-generic) and a prototype (π) interpretation. The copula `is` is ambiguous between a literal and a non-literal (μ) interpretation, as is the transitive verb `drinks`. For good measure, we have added the interpretation of the copula which includes a non-triviality constraint ($\mu, \neg tr$) as in (9). A minimality constraint (10) can be implemented along the same lines. The reader is invited to check that the grammar maps (1) to $\exists x(fox\ x \wedge x = j)$, (7), (9) and (11), i.e. the grammar generates both literal and

non-literal interpretations. It maps (12) to *drink g c* and to (14). As it stands, the grammar overgenerates: it combines the generic reading of the object NP with the literal reading of `is` etc. Such readings can be excluded by features in a more detailed encoding of the fragment. In Appendix A we provide a simple Prolog implementation of the grammar and the syntax – semantics interface following (Pereira and Shieber, 1987) which readers are invited to test.

Conclusion

In the present paper we have developed an approach to metaphor based on standard type theory (a classical higher order logic). We capture an asymmetry between metaphor and simile: the literal interpretation of a metaphor comes out as (mostly) false while its non-literal interpretation is that of a corresponding reduced simile. Our theory captures sense extension in that the property shared between tenor and vehicle includes at least the extension of both. We have presented a compositional syntax – semantics interface, provided a Prolog implementation and sketched a deductive account of resolution. We discussed how the approach addresses issues of generalisation, feature selection, asymmetry, tension, trivialisation, prototypicality, truth conditions, comprehension and generativeness. Summarising in the form of a slogan, our approach can be said to "rescue a weak propositional content of metaphors." To conclude we give our judgement on the commonplace proposition (or metaphor ...) that classical logic, formal semantics and metaphors are uneasy bedfellows: **False**!

Acknowledgements

Many thanks to Carl Vogel, Tony Veale, Ede Zimmermann, Dick Crouch, the two sets of anonymous referees for AISB'99 and Metaphor and Symbol and to John Barnden for stimulating discussion, feedback and support. Any mistakes are my own.

References

E. W. Adams. (1998). A Primer of Probability Logic. CSLI Publications, CSLI Lecture Notes: no. 68.

Aristotle. (1952). Rhetoric. Poetics. In: W.D. Ross (ed.), The Works of Aristotle, Vol. 11, Oxford, Clarendon Press

J. Barwise and R. Cooper, (Ed.). (1981). Generalized Quantifiers and Natural Language. Linguistics and Philosophy, 4, pp.159–219.

M. Black, (Ed.). (1962). Models and Metaphors. Ithaca, NY, Cornell University Press.

G. Carlson and J. Pelletier, (Ed.). (1995). The Generic Book. University of Chicago Press.

A. Church, (Ed.). (1940). A Formulation of the simple Theory of Types. Journal of Symbolic Logic, No.5. pp.65 – 68.

D. Davidson. (1984). What Metaphors Mean. In D. Davidson (Ed.), Inquiries into truth and interpretation (p. 245-64). Oxford: Oxford University Press.

B. Falkenheiner and K. Forbus and D. Gentner. (1989). Structure-Mapping Engine, Artificial Intelligence, 41, pp.1-63

M. Fitting. (1996). First-Order Logic and Automated Theorem Proving, Second Edition. Springer Verlag.

R.J. Fogelin. (1988). Figuratively Speaking, Yale University Press.

L.T.F. Gamut. (1991). Language, Logic and Meaning, Part 1. Chicago University Press, Chicago.

L.T.F. Gamut. (1991). Language, Logic and Meaning, Part 2. Chicago University Press, Chicago.

L. Henkin. (1950). Completeness in the Theory of Types. Journal of Symbolic Logic. Vol 15. pp.81–91.

J. Jaspars. (1994). Calculi for Constructive Communication: A Study of the Dynamics of Partial States. ILLC Dissertation Series 1994-4. Institute for Logic, Language and Computation, Universiteit van Amsterdam, The Netherlands

R. Montague. (1973). The proper treatment of quantification in ordinary English. In J. Hintikka. e.a. (Ed.), Approaches to natural language (pp. 221–242). Reidel.

A. Ortony. (1979). Beyond literal similarity. Psychological Review. 86. pp.161–180.

F.C.N.Pereira and S.M.Shieber. (1987). Prolog and Natural-Language Analysis. CSLI Publications, CSLI Lecture Notes: no. 10.

M. Thomas and D. Mareschal. (1999). Metaphor as Categorisation: A Connectionist Implementation. Proceedings of the AISB'99 Symposium on Metaphor, Artificial Intelligence, and Cognition, 6th–9th April, University of Edinburgh, pp.1-10.

T. Veale and M. Keane. (1992). Conceptual scaffolding: A spatially founded meaning representation for metaphor comprehension. Computational Intelligence, 8(3), 494-519.

C. Vogel. (2000). Dynamic Semantics for Metaphor. This volume.

Appendix A

```
%% meta.pl  A toy DCG implementation, Josef van Genabith, DCU, CA.
%% implication    %% conjunction   %% negation     %% application
```

```
:- op(40,xfy,>).  :- op(30,xfy,&). :- op(20,fy,~). :- op(15,yfx,@).

apply(la(X,Y),X,Y).  %% application & reduction (Pereira & Shieber,1987)

%%

s(S) --> np(NP), vp(VP),     {apply(NP,VP,S)}.

vp(VP) --> v(V), np(NP),     {apply(V,NP,VP)}.

np(la(P,Pj)) --> [john],     {apply(P,john,Pj)}.

np(la(P,Pg)) --> [gasoline], {apply(P,gasoline,Pg)}.

np(la(P,Pc)) --> [my,car],   {apply(P,car,Pc)}.

%% indefinite, then simplified quasi-generic, then prototype reading

np(la(Q,exists(X, fox(X) & Qx))) --> [a,fox], {apply(Q,X,Qx)}.

np(la(Q,forall(X, fox(X) > Qx))) --> [a,fox], {apply(Q,X,Qx)}.

np(la(Q,prty(fox,Q)))            --> [a,fox].

%% first literal, then metaphorical reading

v(la(P,la(X,Sem)))                    --> [is], {apply(P,la(Y,X=Y),Sem)}.

v(la(Q,(la(Y,exists(P,P@Y & QP)))))   --> [is], {apply(Q,la(X,P@X), QP)}.

%% first literal, then metaphorical reading

v(la(Q,la(X,Sem))) --> [drinks], {apply(Q,la(Y,drink(X,Y)),Sem)}.

v(la(Q,la(X,Sem))) --> [drinks],

{apply(Q,la(Y,exists(R,R@Y@X & forall(Z,forall(W,drink(Z,W) > R@W@Z)))),Sem)}.

%%

test :-

  t(N,Sent), s(Sem,Sent,[]), write(N), write(':'), write(' '), write(Sent),

  nl, write('Sem:'), write(':'), write(' '), write(Sem), nl, nl, fail.

test.

  t(1,[john,is,a,fox]).      t(2,[my,car,drinks,gasoline]).

%%
```

The grammar overgenerates. This can be ruled out in terms of features in a more realistic implementation:

```
| ?- test.

1: [john,is,a,fox]  Sem:: exists(X,fox(X)&(john=X))

1: [john,is,a,fox]  Sem:: forall(X,fox(X)>(john=X))

1: [john,is,a,fox]  Sem:: prty(fox,la(X,john=X))

1: [john,is,a,fox]  Sem:: exists(P,P@john&exists(X,fox(X)&P@X))

1: [john,is,a,fox]  Sem:: exists(P,P@john&forall(X,fox(X)>P@X))

1: [john,is,a,fox]  Sem:: exists(P,P@john&prty(fox,la(X,P@X)))

2: [my,car,drinks,gasoline]  Sem:: drink(car,gasoline)

2: [my,car,drinks,gasoline]  Sem:: exists(P,P@gasoline@car&
                                     forall(X,forall(Y,drink(X,Y)>P@X@Y)))
```

Appendix B

To handle example (16): ``I wrestled with the idea'' add the following:

```
np(la(P,Pi)) --> [i],        {apply(P,i,Pi)}.

np(la(P,Pi)) --> [the,idea], {apply(P,idea,Pi)}.

%% first literal, then metaphorical reading

v(la(Q,la(X,Sem))) --> [wrestled,with], {apply(Q,la(Y,wrestle(X,Y)),Sem)}.

v(la(Q,la(X,Sem))) --> [wrestled,with], {apply(Q,la(Y,exists(R,R@Y@X &

forall(Z,forall(W,wrestle(Z,W) > R@W@Z)))),Sem)}.
```

The query responses are as expected:

```
3: [i,wrestled,with,the,idea]  Sem:: wrestle(i,idea)

3: [i,wrestled,with,the,idea]  Sem:: exists(P,P@idea@i&
                         forall(X,forall(Y,wrestle(X,Y)>P@X@Y)))
```

Footnotes

[1]This is the reason why simple meaning postulates (axioms) are of limited use in treatments of metaphor. The problem is the following: consider the metaphorical sentence in (1) above. Assume that it translates as $fox(j)$. Assume further that, for the sake of the argument, we have an axiom stating that all foxes are clever. From these we can deduce $fox(j),\ \forall x(fox(x) \rightarrow clever(x)) \ \vdash \ clever(j)$ as a possible interpretation of (1). This inference is fine even if an additional $human(j)$ axiom is in force. However, things start turning sour as soon as we have another axiom in place that states that the sets of humans and foxes are disjoint: $\forall x \neg(human(x) \wedge fox(x))$. Given this and our previous assumptions, inconsistency strikes: we can prove $human(j) \wedge \neg human(j)$, or indeed any conclusion we wish. The approach developed in the present paper avoids such pitfalls.

[2]The remaining connectives and quantifiers are defined from these in the usual fashion: $\varphi \vee \psi \equiv \neg(\neg\varphi \wedge \neg\psi), \varphi \rightarrow \psi \equiv \neg(\varphi \wedge \neg\psi), \exists x\varphi \equiv \neg\forall x\neg\varphi$.

[3]This example was provided by one of the anonymous reviewers.

[4]The class fox stands proxy for a prototypical individual. $prty$ simply pairs the class with its perceived cultural stereotypes.

[5]Readers unfamiliar with the functional type theory notation may be puzzled by the order of arguments in $R\ g\ c$ in (14). The contribution g of the direct object comes first followed by the contribution c of the subject. In the Prolog implementations in Appendices A and B we switch back to the familiar relational representations: `R(c,g)`.

[6]Mostly confined to jokes.