Structured Meanings in Computational Linguistics

Kees van Deemter
Institute for Perception Research
P.O.Box 513, 5600 MB Eindhoven
The Netherlands.

22 March 1990

1 Introduction

Many natural language processing systems employ truth conditional knowledge representations ('t-representations', etc.) to represent meanings of natural language expressions. T-representations have their strong and their weak sides. A strong side is **logic**: a relation of logical consequence can be defined between such representations. A weak side is **expressive power**: the capacity of t-representations to convey the subtleties of natural language is limited. For instance, let S_L be a sentence that is true on purely logical grounds; then it is predicted that any sentence S is synonymous with "S *and* S_L". This deficiency comes out clearest in propositional attitude constructions, i.e. constructions of the form 'x V that S', where V is an epistemic verb ('knows', 'believes') and S a sentence. Truth conditional accounts of meaning (including intensional ones such as [Montague 1974]) predict wrongly that anybody who knows that S is bound to also know that "S *and* S_L", since the two sentences are t-indistinguishable ([Peters and Saarinen 1982]). The same lack of expressive power dooms, for example, *automatic translation* on the basis of t-representations to failure: t-representations contain only information that is relevant for the truth or falsity of a sentence, dismissing all other information, such as mood, topic-comment structure, etc. ([van Deemter -89]).

This paper investigates a remedy for the expressive poverty of t-representations, namely to let syntactic structure participate in the notion of meaning. This old and persistent idea ([Carnap 1947]), [Lewis 1972], [Cresswell 1985]) was recently taken up in the Rosetta automatic translation program. We will show how Rosetta's concept of meaning overcomes some weaknesses of earlier proposals and how a relation of logical consequence can be defined on top of it.

2 An Old Idea: Structured Meanings

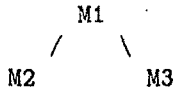
It has been argued that no theory of meaning that is compositional and truth conditional can deal with propositional attitudes. For, whenever two expressions with *different* compositional (syntactic) structures boil down — via the semantic operations connected with their respective structures — to *the same* meaning, a person can fail to see the equivalence: he can carry out the operations in the wrong way, or too slowly ([Cresswell 1985]). Cresswell and others have concluded that syntactic structure has to take part in meaning representations: t-indistinguishable expressions may still have different meanings, due to differences in syntactic structure. D.Lewis, for instance, used semantically interpreted phrase markers (roughly: syntax trees with logical formulas attached to the nodes) as meanings for natural language expressions ([Lewis 1972]). However, this leads to an extremely strict notion of synonymy:

> Perhaps we would cut thereby meanings too finely. For instance, we will be unable to agree with someone who says that a double negation has the same meaning as the corresponding affirmative. ([Lewis 1972])

Also, no relation of *logical consequence* has seen the light for any notion of structured meaning. In the sequel we will deal with the notion of meaning inherent in the Rosetta automatic translation project (e.g. [Landsbergen 1982], [Landsbergen 1985], [Landsbergen 1987], or [de Jong and Appelo 1987]). This notion of meaning — essentially an elaboration of the one proposed by Lewis — allows a suitably *weaker* notion of synonymy, and can also be provided with a notion of *logical consequence*. Thus, some of the weak sides of older "structured meanings" proposals are compensated for.

3 Structured Meanings in Rosetta

Rosetta uses a variant of Montague grammar ([Montague 1974]), in which each syntax rule has a semantic counterpart. Each node in the syntactic derivation tree (D-tree) for a sentence is associated with a semantic rule. Thus, each D-tree is associated with a semantic tree (M-tree), whose nodes are semantic rules and whose leaves are non-logical constants. By applying the semantic rules to their arguments, a logical formula can be calculated for each node in the M-tree that stands for its truthconditional meaning. We will call this formula the *corresponding* formula of the node. Now in Rosetta, a sentence meaning is *not*, as in [Montague 1974], identified with the formula that corresponds to the top node of an M-tree, but with the entire tree. Thus, syntactic structure in Rosetta becomes a part of meaning in much the same way as proposed by D.Lewis (see above). For instance, the English Noun Phrase 'Italian girl' and its Spanish equivalent 'muchacha Italiana' might, if we simplify, both be represented by the same M-tree:



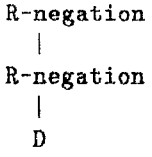
where M2 stands for the sets of Italians, M3 stands for the set of girls and M1 stands for the operation of set intersection. M1 is expressed by different syntax rules in English and Spanish:

> R_1^{Engl}: If α is an Adjective and β is a Noun, then $\alpha\beta$ is a Nom.
> R_1^{Span}: If α is an Adjective and β is a Noun, then $\beta\ \alpha'$ is a Nom, where α' is the adjective α, adjusted to number and gender of the noun β.

By mapping both of these rules onto M1, the two NPs are designated as translations of each other. Now M-trees in Rosetta are used as vehicles for inter-lingual translation, but we will view them as "general purpose" representations for the meanings of natural language expressions. Viewed in this way, the following definition of synonymy (notation: '$\equiv$') between D-trees (and, derivatively, for natural language expressions) is forthcoming:

> **Synonymy** (first version): $D_1 \equiv D_2 \Leftrightarrow_{def}$
> – $D_1 = R_1(a_1, ..., a_n)$ and $D_2 = R_2(b_1, ..., b_n)$, where R_1 and R_2 map onto the same meaning rule, and where it holds for all $1 \leq i \leq n$ that $a_i \equiv b_i$, or
> – D_1 and D_2 are basic expressions which map onto the same basic meaning.

(Definition of synonymy for M-trees, at this stage, comes down to simple equality of the trees.) This notion of meaning takes syntactic structure into account, but does not "cut meanings too finely", since any two linguistic constructions can be designated as synonymous. For instance, Lewis' "double negation" problem can be countered as follows: the syntax rules of double negation ($R_{doubleneg}$) and plain affirmation (R_{affirm}) can be mapped onto one and the same meaning rule, if the grammar writer decides that they are semantically indistinguishable. Alternatively, the semantic relation between a D-tree of the form



and its constituent tree D may be accounted for if both trees are mapped onto one and the same M-tree. Effectively, this would come down to an extension of Rosetta with "rules of synonymy" for entire trees, rather than for individual syntax rules.

4 Inference with M-trees

Arguably, our grip on the notion of meaning is incomplete if only the limiting case of structural equivalence is dealt with, leaving aside the more general case of structural consequence ($\models_M$). Under what conditions does, for instance, one belief *follow* from another? Rosetta's isomorphy-based notion of meaning seems ill-equipped to deal with inference, but we claim that an extrapolation is possible.

A natural boundary conditions on $\models_M$ is logical validity: no rule may lead from true premises to a false conclusion. Writing '$\models$' for the relation holding between M-trees if the formulas corresponding to their top-nodes stand in the relation of logical consequence, this gives:

> **Validity:** $T_a \models_M T_b$ only if $T_a \models T_b$.

Given validity as an upper bound, we seek reasonable lower bounds on structural inference. It is not generally valid to allow that a tree has all its *subtrees* as structural consequences. (For instance, the

negation of a tree T does not have the subtree T as a consequence.) However, a solution can be found if we take the dual nature of our concept of meaning into account: M-trees combine structural and logical information. Therefore, if one tree is a subtree of another tree, and also a purely logical consequence of the bigger tree, then the inference is indisputable; for the inference is logically correct and there can be no difference in syntactic structure:

> **Subtree Principle** (1^{st} version): If (i) $T_a \models T_b$ and (ii) T_b is a subtree of T_a then $T_a \models_M T_b$.

However, we have to exclude as "pathological" cases all those situations in which it is not one and the same subtree T_b that takes care of the logical and the structural side: we cannot allow inferences such as the following — where $\tilde{S}$ abbreviates a "paraphrase" of S, namely a sentence that is logically, but not structurally, equivalent to S (see below) — even though they fulfil both conditions of the **Subtree Principle**:

(1) $(S \vee \neg S)$ & $\tilde{S} \models_M S$
(2) $(\neg S \rightarrow \tilde{S}_1)$ & $(S \rightarrow (S_1 \vee S_2)) \models_M S_1 \vee S_2$.

These inferences are not structurally valid, given the structural differences between the conclusion and the relevant part of the premise. Let an atomic sentential fragment (*asf*) be a sentential M-tree no proper part of which is sentential itself. To be on the safe side, we might forbid that "paraphases" of asf's from the conclusion occur in the premisse:

> **Subtree Principle** (2^{nd} version): *If* (i) $T_a \models T_b$ and (ii) T_b is a subtree of T_a and (iii) If T_1 is an asf that occurs essentially in T_a and T_2 is an asf that occurs essentially in T_b, then T_1 is not a paraphrase of T_2, *then* $T_a \models_M T_b$,

where a *paraphrase* is a logical equivalent that falls short of structural equivalence:

> **Paraphrase** (1^{st} version): T_1 is a paraphrase of $T_2 \Leftrightarrow_{Def} T_1 \models T_2$ and $T_2 \models T_1$ but none of the two is a subtree of the other.

The resulting logic is quite uncommon unless stronger lower bounds are given. For instance, if (ii) is a necessary condition, there cannot be any tree T such that $\models_M$ T. Consequently, the Deduction Theorem will not hold. Also, if (iii) is a necessary condition, then Conjunction Elimination fails to hold. In fact, it holds for all S_1 and S_2 that $S_1 \& \tilde{S}_2 \not\models_M S_2$. To remedy this defect, (iii) may be weakened to allow logically *inessential* occurrences of paraphrases:

> **Inessential occurrence**: An occurrence of T in the premisse (conclusion) of an inference is *inessential* if the inference goes through when T is replaced by an arbitrary T′ everywhere in the premisse (conclusion).

For instance, the occurrence of $\tilde{S}$ in (1) and (2) is essential, but its occurrence in $S \& \tilde{S} \models_M S$ is inessential and therefore harmless. As a result of this change, a restricted version of Conjunction Elimination holds, to the effect that a conjunction will structurally imply any of its conjuncts, *provided* the conclusion conjunct does not contain two asf's that are paraphrases of eachother. This concludes our formalization of the "subtree" intuition. If we want to cover more ground, we need a more liberal concept than the structural notion of one tree being a subtree of another. First, a more subtle structural notion may be employed. For instance, an inference from *Each dog barks loudly* to *Each black dog barks* must be allowed, it seems, even though none of the two M-trees is a part of the other. Therefore, a relation of constituent-wise comparability ($\approx_c$, definition follows) is called for. It is important to note that the "direction" of the comparison (which of the two subsumes which) is irrelevant, since the *logical* requirement (i) determines the direction of the inference:

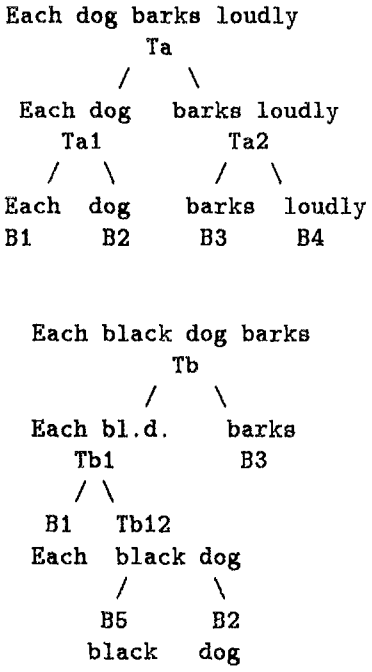
> **Subtree Principle** (3^{rd} version): If (i) $T_a \models T_b$ and (ii) $T_a \approx_c T_b$ and (iii) *(as above)*, then $T_a \models_M T_b$.

If the notation $\approx$ stands for the symmetrical relation that holds between two trees if one of them is a subtree of the other, this is the definition of the relation $\approx_c$:

> **Comparability**: $T_a \approx_c T_b \Leftrightarrow_{Def}$ $\exists m, n > 0$ such that $T_a = < T_{a1},...,T_{an} >$ and $T_b = < T_{b1},...,T_{bm} >$, where either $\forall\, T_{ai}\ \exists\, T_{bj} : T_{ai} \approx T_{bj}$ or $\forall\, T_{bj}\ \exists\, T_{ai} : T_{bj} \approx T_{ai}$.

Here, $T_u = < T_{u1},...,T_{un} >$ means that T_u can be decomposed (at an arbitrary level of the tree) as the sequence $T_{u1},...,T_{un}$.

Example: The M-trees for *Each black dog barks* and *Each dog barks* stand in the relations $\approx_c$ *and* $\approx$, while the M-trees for *Each dog barks loudly* and *Each black dog barks* do not stand in the relation $\approx$, but they *do* stand in the relation $\approx_c$. They are constituent-wise comparable, so since the first logically implies ($\models$) the second, the first must also have the second as a structural consequence ($\models_M$):



Here, $T_a \approx_c T_b$ holds, for $T_a = < B_1, B_2, T_{a2} >$, and $T_b = < B_1, T_{b12}, B_3 >$, while B_2 is a subtree of T_{b12} and B_3 is a subtree of T_{a2}. **End of Example**

Note that, by replacing the subtree notion by the symmetrical notion $\approx_c$, we now allow a conclusion to introduce asf's that do not occur in the premisse. For instance, under appropriate assumptions, it will hold that S $\models_M$ S *and* S_L, for logically true S_L. This defect can be remedied simply if we add a clause that prevents a conclusion from containing any novel asf's (see (iv), below).

So far, the Subtree Principle still formalizes a strictly structural approach. But there ought to be more than that. In the ideolect of a given language user, two grammar rules, or two lexical items, may be semantically related without any strictly structural notion being involved. Within the bounds of Validity and the Subtree Principle, the grammar writer is free to designate certain pairs of syntax rules or lexical items as semantically related. Since, again, the direction of the relation is irrelevant, this refinement can easily be built in into the definition of $\approx_c$. If this is done, the relation $\approx_c$ will also hold between *Each mammal barks loudly* and *Each black dog barks*, assuming that 'mammal' and 'dog' are semantically related. Note, however, that these stipulations need not be the same for all language users: different stipulations of structural relatedness may reflect differences in linguistic competence ([Partee 1982]). In short, our proposal implements the hypothesis that structural relations hold for everyone, while linguistic relations allow individual variation.

If all the suggested improvements on the Subtree Principle are taken into account, one might venture the following definition of structural consequence:

> **Subtree Principle**(final version): $T_a \models_M T_b \Leftrightarrow_{Def}$ (i) $T_a \models T_b$ and (ii) $T_a \approx_c T_b$ and (iii) If T_1 is an asf that occurs essentially in T_a and T_2 is an asf that occurs essentially in T_b, then T_1 is not a paraphrase of T_2, and (iv) all asf's of T_b occur in T_a.

Since the notion of a subtree has now been replaced by constituent-wise comparability, the notion of a paraphrase must be redefined:

> **Paraphrase** (final version): T_1 is a paraphrase of $T_2 \Leftrightarrow_{Def} T_1 \models T_2$ and $T_2 \models T_1$ but $T_1 \not\approx_c T_2$.

Assuming that a notion of inference has been established along these lines, *synonymy* between M-trees can now be defined as mutual structural consequence (synonymy of D-trees is analogous):

> **Synonymy** (final version): T_1 and T_2 are synonymous $\Leftrightarrow_{Def} T_1 \models_M T_2$ and $T_2 \models_M T_1$.

If the clauses in the first or the second version of the Subtree Principle are taken as collectively sufficient and necessary, the defined notion of synonymy coincides with the original Rosetta notion of "having the same M-tree". (In this case, $T_a \models_M T_b$ and $T_b \models_M T_a \Leftrightarrow T_a = T_b$.) This conveniently simple situation breaks down in later versions of the Subtree Principle, where the relation of constituent-wise comparability is used. A simple example:

> (a) *John walks and John walks slowly*, and
> (b) *John walks slowly and John walks.*

The Subtree Principle (3^{rd} or final version) implies that $(T_a \models_M T_b)$ & $(T_b \models_M T_a)$, and therefore, (a) and (b) are predicted to be synonymous, despite the difference between their corresponding M-trees — which would have made them nonsynonymous in Rosetta's original notion of synonymy.

5 Applications and Limitations

In section 4, we presented one among several possible ways in which a notion of structural consequence can

be defined on the basis of Rosetta's M-trees. Now we will indicate briefly how M-trees can be applied to propositional attitudes and to natural language generation outside the context of automatic translation. But, there is a *caveat*, discussed under the header of "mixed inference".

Propositional Attitudes. Given that meanings are M-trees, the natural solution to the problem of "de dicto" propositional attitudes is to let epistemic attitudes denote a relation between an individual and an M-tree. Consequently, if a person x knows that S, while S' and S share the same M-tree, then x is predicted to also know that S'. Since this amounts to a much stronger relation of synonymy than logical equivalence (t-equivalence), the problems noted in the introduction do not arise. The general situation is that if x knows that S and S $\models_M$ S'', then it is predicted that x also knows that S''.

Natural Language Generation. Even outside the domain of automatic translation, M-trees can be used for natural language generation. For example, in a natural language question-answering application, the M-tree derived from the input-question can serve as a basis for generation, after some operations on the original M-tree, in which a yes-no question is changed into an affirmative or negative answer, for instance. In most applications where there is no M-tree available, other means than M-trees can be used. For instance, when the user of a query system asks assistance from the computer's help facility, pre-stored natural language text can replace M-trees.

Mixed Inference. We have seen that inference on the basis of M-trees is feasible, but how about inference on the basis of premises, some of which are purely logical while others are fully dressed M-trees? Two obvious approaches (where $\models_x$ denotes mixed inference, and T_ϕ is a variable over those M-trees having ϕ corresponding to their top nodes) are

(i) $\phi,\ T \models_x \psi \quad \Leftrightarrow_{def} \quad \phi\ , \chi \models \psi$ (χ is the corresponding formula of T's top node),

(ii) $\phi,\ T_1 \models_x T_2 \ \Leftrightarrow_{def}\ \forall\, T_\phi : T_\phi, T_1 \models_M T_2$.

Neither solution is satisfying: the first leaves T's linguistic structure unused; and the second, which quantifies over all the possible ways in which ϕ can be expressed, is computationally intractable. The problem with mixed inference illustrates the one weak side of structured meanings: they let linguistic structure contribute to the meaning of an expression, but it is impossible to say (in model theoretic terms) *what* it contributes.

6 References

[**Carnap 1947**] Carnap,R. Meaning and Necessity. University of Chicago Press, Chicago.

[**Cresswell 1985**] Cresswell, M.J. Structured Meanings. The Semantics of Propositional Attitudes. MIT Press, Cambridge,Mass.

[**van Deemter -89**] Structured Meanings Revisited, IPO Manuscript 693-II, to appear in Bunt and van Hout (eds.) Language Technology. Foris, Dordrecht.

[**de Jong and Appelo 1987**] de Jong,F. Appelo,L. Synonymy and Translation, 1987 In: Proceedings of the 6th. Amsterdam Colloquium.

[**Landsbergen 1982**] Landsbergen,J. Machine Translation Based on Logically Isomorphic Montague Grammars. Proceedings COLING 1982.

[**Landsbergen 1985**] Landsbergen,J. Isomorphic Grammars and their use in the ROSETTA Translation System. Philips Research M.S. 12.950. In King,M. (ed.), Machine Translation: the state of the art. Edinburgh University Press.

[**Landsbergen 1987**] Landsbergen,J. Montague Grammar and Machine Translation, in Whitelock,P. et al. (eds.) Linguistic Theory and Computer Applications, Acad. Press, London.

[**Lewis 1972**] General Semantics, in D.Davidson and G.Harman (eds.) Semantics of Natural Language. Reidel, Dordrecht.

[**Montague 1974**] Montague,R. The Proper Treatment of Quantification in Ordinary English. In R.H.Thomason (ed.), Formal Philosophy. Yale University Press, New Haven and London.

[**Partee 1982**] Partee,B. Belief Sentences and the Limits of Semantics. In Peters and Saarinen (eds.)

[**Peters and Saarinen 1982**] Peters,S. and Saarinen,E. (eds.) Processes, Beliefs and Questions, D.Reidel, Dordrecht.