HARRY BUNT AND REINHARD MUSKENS

COMPUTATIONAL SEMANTICS

1. Introduction

Computational semantics is concerned with computing the meanings of linguistic objects such as sentences, text fragments, and dialogue contributions. As such it is the interdisciplinary child of semantics, the study of meaning and its linguistic encoding, and computational linguistics, the discipline that is concerned with computations on linguistic objects.

From one parent computational semantics inherits concepts and techniques that have been developed under the banner of formal (or model-theoretic) semantics. This blend of logic and linguistics applies the methods of logic to the description of meaning. From the other parent the young discipline inherits methods and techniques for parsing sentences, for effective and efficient representation of syntactic structure and logical form, and for reasoning with semantic information. As is often the case with inheritance, blessings are mixed and computational semantics' legacy not only contains useful elements but also many problems and unresolved issues. In fact, many of the well-known problems in semantics such as those relating to handling ambiguity and vagueness, as well as fundamental issues regarding compositionality and context-dependence, re-appear in computational semantics in a form even more compelling than in formal semantics. In this introductory chapter we shall highlight some of the issues and problems we think are most pressing. We shall start with discussing a straightforward way to implement formal semantics in the tradition that started with Montague's work. Then, after a conceptual analysis of the principle of compositionality and a discussion of its role in computational work, the *problem of ambiguity* will be introduced and it will be shown that the straightforward way of taking standard formal semantics as a specification of computer algorithms needs to be amended. An important issue in computational semantics is that of the *context dependency* of language. The task set to computational semanticists is to provide concrete linguistic objects with abstract meanings; as concrete linguistic objects are utterances rather than sentences (types of utterances), computational semantics

needs to draw in a lot of the work that takes context into account and traditionally goes under the banner of pragmatics. After a discussion of this, and of the need for *underspecified representations* of meaning, which arises out of the ambiguity problem, the chapter closes with an overview of the other chapters in this book.

2. Algorithmic realisations of model-theoretic semantics

The beginnings of computational semantics can be traced back to the pioneering work of Richard Montague in formal semantics. His work is of direct relevance to computational semantics, due to the detail and precision with which it describes how the expressions of a small 'fragment' of some natural language can be associated with meaning representations in a formal, logical language for which a model-theoretic interpretation is provided.

The association of language with logic is defined in two steps in Montague's work. First, since natural language is ambiguous but logic is not, the expressions of the fragment under consideration (let us call it NL_0) must be 'disambiguated'; each expression may have one or more *readings*. Formally this boils down to defining a 'disambiguation' relation ρ between expressions in NL_0 and expressions in some language $\mathcal{R}$ of readings. In Montague (1973) $\mathcal{R}$ is a collection of 'analysis trees', but we may illustrate the set-up here by letting $\mathcal{R}$ be a collection of Logical Forms in the sense of May (1977). In (1) below, the Logical Forms (1c) and (1d) are readings of (1a), while (1b) corresponds to the constituent structure of (1a). Accordingly we have that (1a)ρ(1c) and that (1a)ρ(1d).

(1) a. Every student caught a fish

b. $[_S [_{NP}$ every student$]$ $[_{VP}$ caught $[_{NP}$ a fish$]]]$

c. [[every student]^1 [[a fish]^2 [e_1[caught e_2]]]]

d. [[a fish]^2 [[every student]^1 [e_1[caught e_2]]]]

e. $\forall x[S(x) \rightarrow \exists y[F(y) \wedge C(x,y)]]$

f. $\exists y[F(y) \wedge \forall x[S(x) \rightarrow C(x,y)]]$

An important constraint on the language $\mathcal{R}$ in Montague's set-up is that its elements should be *uniquely readable* and that it should be

possible to define a translation *function* τ from $\mathcal{R}$ into some interpreted logical language.[1] In classical Montague Semantics the logic of choice is Intensional Logic (IL), a variant of Russell's and Church's Theory of Types. Here we may illustrate the procedure by using classical logic, assuming that $\tau((1c)) = (1e)$ and that $\tau((1d)) = (1f)$.

Once ρ and τ are defined, we have an interpretation of NL_0, provided that the target of τ really is an interpreted logic. For such logics $\mathcal{L}$ an interpretation function $[\![.]\!]$ is defined which, given a model M and an assignment a, associates with each well-formed expression φ of $\mathcal{L}$ a semantic value $[\![\varphi]\!]^{M,a}$ constructed from the entities in M. Obviously, the compound relation $\rho \circ \tau \circ \lambda\varphi.[\![\varphi]\!]^{M,a}$ associates semantic values with expressions of NL_0. Montague stressed that the intermediate stage of logical representations is inessential: in theory it is possible to map readings of natural language expressions to model-theoretic meanings directly.

Suppose that ρ and τ are defined for some fragment NL_0 and that sentences in NL_0 are thus associated with truth-conditions. Then the following kind of questions are answerable in principle.

1. Does sentence S on its reading r **follow** from sentence S' on reading r'? This boils down to asking whether $\tau(r') \models_{\mathcal{L}} \tau(r)$ (where $\models_{\mathcal{L}}$ is the relation of logical consequence in the logic $\mathcal{L}$).
2. Is sentence S on its reading r **true** in situation s? This boils down to finding a model M which models the relevant aspects of s and to asking whether $M \models \tau(r)$.

This means that notions of *truth* and *logical consequence* are now defined for NL_0, be it that both are given relative to the various possible readings of the expressions involved, a restriction which is not unnatural.

There is some empirical bite to this. People have *intuitions* about the notions of truth and consequence in their native language, even though these intuitions usually leave many cases undecided. True, it is often difficult to elicit clear judgements about the question whether some sentence is true in a given situation or the question whether one sentence follows from another, and even speakers of the same ideolect may not find themselves in agreement about some of these questions. But there are also many cases in which judgements are clear. This means that the entailment relation and the notion of truth that we

[1] τ should be functional *modulo* logical equivalence, i.e. if $\tau(\alpha) = A$ and A' is logically equivalent with A, we also write $\tau(\alpha) = A'$.

get from ρ and τ may in fact be contradicted by empirical evidence, if we accept native speaker judgement as such. Montague thought of his work as pertaining purely to mathematics, but in the light of the possibility that a particular choice of ρ and τ may be falsified, we may re-interpret his theory as being empirical and start playing the usual game of conjectures and refutations. By considering larger and larger fragments NL_0, adapting the definitions of ρ and τ, playing with various possibilities for the intermediate language $\mathcal{R}$, and varying the logic $\mathcal{L}$, we may try to obtain better and better approximations to natural language, its truth conditions, and the natural language entailment relation.

Turning to the possibility of realising Montague Grammar algorithmically, we see that for many steps existing technology can be used. Parsing technology provides us with a realisation of the transition from strings to surface structures such as the one from (1a) to (1b). And once we have found logical representations such as the ones in (1e) and (1f), theorem proving technology can be used to provide us with a partial answer to the entailment question discussed in 1. above. Since any logic $\mathcal{L}$ that aspires to translate natural language will be undecidable, we can only hope for a partial answer here, but in practice a lot of reasoning can be automated and many theorem provers (especially those of the tableau variety) also as a bonus provide us with counterexamples to invalid arguments. Evaluation of a sentence on some given model, as in 2., can also be automated, as the excellent educational program *Tarski's World* amply illustrates. A restriction here is that models must be finite (otherwise quantification cannot be decided). In practice, models can take the concrete form of databases. Some of the earliest work in this area are that in the PHLIQA project at Philips Research (see Bronnenberg et al., 1979; Bunt, 1981; Scha, 1983) and that of Konolige (1986).

This leaves us with the transitions from surface structure to Logical Form, e.g. from (1b) to (1c) or (1d), and from Logical Form to $\mathcal{L}$, e.g. from (1c) to (1e). As we are now in the province of computational semantics proper, let us look at these transitions in some detail. Here is a little calculus that performs the first of the tasks mentioned. It uses a *storage* mechanism inspired by that of Cooper (1983) and computes a ternary relation κ between (a) a surface structure, (b) a Logical Form, and (c) the store, which will be a set of superscripted surface structures of category NP here.

(2) (Terminate) $\kappa(A, \alpha, \emptyset)$, if A is a terminal;

(Store) $\kappa(NP, e_k, \{NP^k\})$, where k is fresh;

(Pass) $$\frac{\kappa(A, \alpha, s) \quad \kappa(B, \beta, s')}{\kappa([_C AB], [\alpha\beta], s \cup s')}$$

(Retrieve) $$\frac{\kappa(NP, \nu, s) \quad \kappa(S, \sigma, s' \cup \{NP^k\})}{\kappa(S, [\nu^k \sigma], s \cup s')}$$

In the Pass rule, C can be any syntactic category; in the Store and Retrieve rules, NP and S stand for any surface structure of category NP and S, respectively. The idea is that a Logical Form can be obtained from a given surface structure by traversing the latter in a bottom-up manner, putting elements that can raise, such as quantified NPs, in store, and retrieving them later nondeterministically at some S node.[2] Applied to (1), for example, the calculus can be used to derive $\kappa((1b), (1c), \emptyset)$. Rule Store licences the NPs *every student*, with Logical Form e_1 and store $\{[_{NP}$ every student$]^1\}$, and *a fish* with logical form e_2 and store $\{[_{NP}$ a fish$]^2\}$. Rule Pass can be used to derive the VP *caught a fish* (where *caught* is licensed by rule Terminate) with Logical Form [caught e_2], and can subsequently be used to obtain *every student caught a fish* with Logical Form $[e_1$[caught e_2]]. Rule Pass also licenses *a fish* and *every student* with Logical Forms [a fish]^2 and [every student]^1, respectively; finally, successive applications of Retrieve result in the derivation of $\kappa([_S[_{NP}$ every student] $[_{VP}$ caught $[_{NP}$ a fish]]], [[a fish]^2 [e_1[caught e_2]]],$\{[_{NP}$ every student$]^1\}$), and to the desired end result $\kappa([_S[_{NP}$ every student] $[_{VP}$ caught $[_{NP}$ a fish]]], [[every student]^1[[a fish]^2 [e_1[caught e_2]]]], $\emptyset$).

The calculus in (2) is simple, but it has a nice property. It is perhaps worth noting that in cases such as (3) below, where an NP can raise out of NP, the latter can happen only if the embedded NP is retrieved from store later than the embedding one. This means that it will not be possible to derive forms where *a company* does not c-command its trace.

[2] Note that the calculus allows retrieving an NP and immediately storing it by using $\kappa(NP, e_j, \{NP^j\})$ as the left premise in the (Retrieve) rule. This will lead to a structure $[e_j^k \sigma]$, with NP^j stored and σ containing e_k. Further processing would lead to a structure $[NP^j[\cdots[e_j^k[\cdots e_k \cdots]]\cdots]]$, i.e. to a chain. At present structures $[e_j^k \sigma]$ will be left uninterpreted by the (Quantify In) rule discussed below, but the addition of a simple type-lifting rule would render them interpretable.

(3) Every representative of a company saw some samples

Our relation ρ can be defined by letting $\rho(A, \alpha)$ if and only if $\kappa(A, \alpha, \emptyset)$ and it may be observed that this in fact gives a computational realisation of the disambiguation relation, as the calculus above is almost identical to the logic program implementing it.

The stage is now set for translating Logical Forms into logical formulas. One of the central ideas in Montague Grammar is that this can be done by shifting to a *typed* logic with lambda abstraction over variables of arbitrary type. Translations of terminal elements can then be as in the following mini-lexicon.

(4)	every	$\rightsquigarrow \lambda P' \lambda P \forall x[P'(x) \rightarrow P(x)]$	(type $(et)((et)t)$)
	a	$\rightsquigarrow \lambda Q' \lambda Q \exists y[Q'(y) \wedge Q(y)]$	(type $(et)((et)t)$)
	student	$\rightsquigarrow S$	(type et)
	fish	$\rightsquigarrow F$	(type et)
	caught	$\rightsquigarrow C$	(type $e(et)$)

The mapping τ can be defined as follows.

(5) (Lex) $\tau(\alpha) = A$, if $\alpha \rightsquigarrow A$;
$\tau(e_k) = x_k$, (type e)

(App) $$\tau([\alpha\beta]) = \begin{cases} \tau(\alpha)(\tau(\beta)), & \text{if } \tau(\alpha)(\tau(\beta)) \text{ is well-typed,} \\ \tau(\beta)(\tau(\alpha)), & \text{if } \tau(\beta)(\tau(\alpha)) \text{ is well-typed,} \\ \text{undefined otherwise;} \end{cases}$$

(Q In) $$\tau([\nu^k \sigma]) = \begin{cases} \tau(\nu)(\lambda x_k.\tau(\sigma)), & \text{if this is well-typed,} \\ \text{undefined otherwise.} \end{cases}$$

Here α, β and ν range over labeled bracketings which do not carry an outermost superscript, so that (App) does not apply when (Q(uantify) In) does. Clearly, maximally one of the first two cases in (Apply) will lead to results on any given input. The reader will note that our previous requirement that $\tau((1c)) = (1e)$ and that $\tau((1d)) = (1f)$ is met.[3]

[3] Applying rule Q In twice for processing the Logical Form parts [every student]^1 and [a fish]^2, and using rule App (plus Lex) to translate the resulting subexpressions, results in the formula $(\lambda P \forall x[S(x) \rightarrow P(x)])(\lambda x_1.(\lambda Q \exists y[F(y) \wedge Q(y)])(\lambda x_2.(C(x_2)(x_1))))$, which is a notational variant of (1e).

We are now in the possession of a well-defined disambiguation relation ρ and a well-defined translation function τ, both easily implementable. But there is an obvious inefficiency in the combination of our calculi, since while ρ is computed by recursing over phrase structures, the function τ is computed by recursing again over the output of ρ. It would be nicer to have a calculus that computes $\rho \circ \tau$ in one fell swoop. Here is one.

(6) (Terminate) $\vartheta(A, A', \emptyset)$, if $A \rightsquigarrow A'$ is in the lexicon;

(Store) $\vartheta(NP, x_k, \{NP^k\})$, where k is fresh;

(Apply) $$\frac{\vartheta(A, A', s) \quad \vartheta(B, B', s')}{\vartheta([_C AB], A'(B'), s \cup s')};$$
$$\frac{\vartheta(A, A', s) \quad \vartheta(B, B', s')}{\vartheta([_C AB], B'(A'), s \cup s')};$$

(Retrieve) $$\frac{\vartheta(NP, Q, s) \quad \vartheta(S, S', s' \cup \{NP^k\})}{\vartheta(S, Q(\lambda x_k.S'), s \cup s')}$$

Again we have the self-evident side conditions that the second argument of ϑ must remain well-typed. Clearly $\vartheta(A, A', s)$ will be derivable for some labelled bracketing A, term A', and store s if and only if there is a Logical Form α such that $\kappa(A, \alpha, s)$ and $\tau(\alpha) = A'$ are derivable in our previous calculi. Since A and A' are in the relation $\rho \circ \tau$ iff $\vartheta(A, A', \emptyset)$, we have a direct computation of that relation. Intermediate computation of Logical Forms is avoided. The calculus in (6) is very close to Cooper's original mechanism of Quantifier Storage, be it that it stores NP surface structures, not NP meanings. It has perhaps some edge over existing systems of Cooper storage. On the one hand it repairs the difficulty that Cooper's original formulation had with sentences such as (3), which contain an NP that can rise out of NP. On the other hand the calculus keeps the structure of stored elements simple. The mechanism of 'Nested Cooper Storage' in Keller (1986) is closely related to the present proposal, but Keller's 'nested stores' are rather more complex data structures than our sets of labelled bracketings for NPs.

This closes the gap between the output of our parser, which we assumed to be in tree format, and the input of our theorem prover, which we took to be classical logic. The calculus in (6) will work for toy context-free grammars of natural language fragments, but will also scale up to more realistic systems.

At this point it may seem that implementation of semantic theories is plain sailing. In order to automatically get information about entailments in natural language, parse the sentences involved, translate the resulting constituent structures to some logical language using a mechanism such as that in (6), and then use standard theorem proving technology. (Less than standard theorem proving technology will be needed of course if your target logic is less than standard.) The procedure seems simple enough, but in fact it cannot be realised on any realistic scale. The reason is the *pervasive ambiguity of natural language*. This will be explained in section 4 below.

3. Compositionality

3.1. *Compositionality and contextuality*

Montague's work falls squarely within the tradition of semantics following the Principle of *Compositionality*, which says in its most general formulation that *The meaning of a compound expression is a function of the meanings of its parts.* Partee et al. (1990) formulate the principle slightly more restrictively as follows: *The meaning of a compound expression is a function of the meanings of its parts and of the syntactic rule by which they are combined.*

Montague applies the compositionality principle on a rule-to-rule basis, assuming that for each syntactic rule, specifying how an expression can be built from simpler ones, the grammar contains a corresponding semantic rule that says how the meaning of the expression depends on the meanings of the parts.

It should be noted, however, that Compositionality in its most general formulation does not necessarily require a rule-to-rule correspondence between syntax and semantics. The notion of 'part', occurring in the principle, is often understood as 'constituent' in the sense of a substructure that has a significance in a syntactic structural description, but this is an unnecessarily restricted interpretation. A grammar may define the set of well-formed expressions of a language by means of derivation rules without attributing a structural syntactic significance to the elements that are used in the rules (as in some versions of Categorial Grammar); the semantic composition function may then operate on these 'parts' rather than on syntactically significant parts.

Intuitively, it would seem that compositionality is a strong property of the way syntax and semantics are related in a grammar, but this is

less obvious when a notion of 'part' is allowed that does not necessarily have a syntactic significance. And even when only syntactically significant parts are allowed, the power of Compositionality is not really clear, since the notion of syntactic part is theory-dependent; different grammatical theories use different kinds of parts. Janssen (1996) has shown that many purported counterexamples to Compositionality in sentence meaning can be handled by introducing new kinds of part, new syntactic constructions, or new kinds of meaning. Similarly, he proves that for any recursively enumerable language L (i.e. any language that can be generated by a Turing machine) and any computable function M that assigns meanings to the expressions of L, it is possible to cast M in a compositional form. This latter result strongly suggests that any grammar for any given natural language can be cast in a compositional form. In other words, if natural languages can be described by grammars at all, then they can be described by compositional grammars.

In view of these results, one may feel that the compositionality principle does not really make a strong claim about the relation between syntax and semantics. We feel that this is indeed so and agree with Janssen in that the principle merely serves a methodological purpose, not strictly an empirical one. But there is a claim that is often felt to be entailed by Compositionality, although in fact it is much stronger. This is the view that the meaning of an expression can only depend on the expression itself and thus entails the negation of Frege's principle of *Contextuality*, which says that *"One should ask for the meaning of a word only in the context of a sentence, and not in isolation."* That Compositionality in fact does not entail the negation of Contextuality (*pace* Janssen (1996)) is easily seen, however, as the first principle literally only constrains the relation between the meanings of complex expressions and the meanings of their parts and is silent about the question where simple expressions derive their meanings from.

Let us consider some examples to show that Compositionality and Contextuality can peacefully coexist. Phenomena such as anaphora and VP deletion involve the dependence of expressions upon other expressions. For example, on the intended reading of (7a) the pronouns *"they"* and *"it"* depend on previous material for their interpretation. *"They"* refers back to the five girls and *it* refers back to the kissing. This means that these words cannot be taken to have meaning in isolation. On the other hand, once the simple expressions in (7a) are provided with their right interpretation, it seems obvious that the meanings of mother categories can be computed from the meanings of their daughters. So in (7a) words do not have meaning in isolation but the meaning

of complex expressions are functions of the meanings of their parts. Similarly, the interpretation of *"did"* in (7b) cannot be considered in isolation, as it obviously depends on the previous VP. But once *"did"* is interpreted correctly as *kissed five girls* the meaning of the sentence can be computed in a rule-by-rule way.

(7) a. John kissed five girls. They liked it.

b. John first kissed five girls and then Max did

We also want to point out that the question in how far expressions can be considered in isolation, depends on the notion of 'expression' that we are using. Are we talking about the type or the token, the sentence or the utterance? The tokens in (8a) and (8b) are different occurrences of the same sentence.

(8) a. Where *does* Mary live?

b. WHERE DOES MARY LIVE

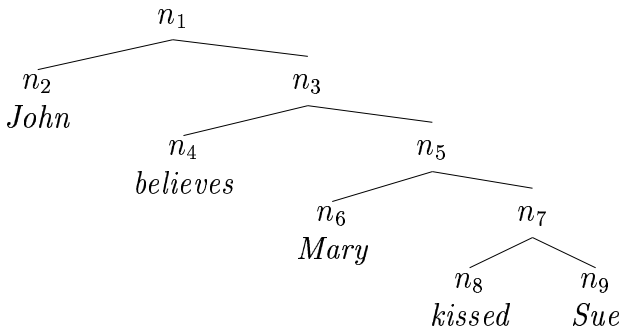
A *sentence* being a linguistic abstraction from utterances, there can be no methodological objection against assuming its meaning to be some sort of abstraction of the meanings of all its possible utterances in different contexts. This gives a certain independence from context, be it an imperfect one. *Utterances* on the other hand, are clearly context-dependent to a large degree. The meaning of an utterance depends not only on the information contained in the utterance itself, but is also strongly dependent on contextual considerations. This consideration is the more pressing in computational semantics as the task of the computational semanticist will often consist in devising algorithms to provide concrete utterances with a precise meaning.

3.2. *Compositionality, bottom-up construction, and bottom-up evaluation*

There are a number of ideas closely related to Compositionality that should carefully be distinguished from the concept itself. We discuss two of them here as they are relevant to the computational perspective on semantics. The first we may call the idea of *Bottom-up Construction*, the second may be dubbed *Bottom-up Evaluation*. In order to explain what Bottom-up Construction is, we have drawn a syntactic tree for the sentence *"John believes Mary kissed Sue"* in (9). In (10) the tree in (9) gets interpreted by means of a system of equations, each equation

expressing the value $\sigma(n_i)$ of node n_i, either in terms of some constant (if n_i is a leaf node) or in terms of the values of its daughters.

(9)



(10)
$$\begin{aligned}
\sigma(n_2) &= \mathit{john}\\
\sigma(n_4) &= \mathit{bel}\\
\sigma(n_6) &= \mathit{mary}\\
\sigma(n_8) &= \mathit{kiss}\\
\sigma(n_9) &= \mathit{sue}\\
\sigma(n_1) &= \sigma(n_3)(\sigma(n_2))\\
\sigma(n_3) &= \sigma(n_4)(\sigma(n_5))\\
\sigma(n_5) &= \sigma(n_7)(\sigma(n_6))\\
\sigma(n_7) &= \sigma(n_8)(\sigma(n_9))
\end{aligned}$$

From the equations in (10) the value of $\sigma(n_1)$ can be computed to be $\mathit{bel}(\mathit{kiss}(\mathit{sue})(\mathit{mary}))(\mathit{john})$. But note that there are different ways to go about here. A *top-down* construction could first compute $\sigma(n_1) = \sigma(n_3)(\sigma(n_2))$, then $\sigma(n_1) = \sigma(n_4)(\sigma(n_5))(\sigma(n_2))$, then $\sigma(n_1) = \sigma(n_4)(\sigma(n_7)(\sigma(n_6)))(\sigma(n_2))$, etc. But working *bottom-up* we could first derive $\sigma(n_7) = \mathit{kiss}(\mathit{sue})$, then $\sigma(n_5) = \mathit{kiss}(\mathit{sue})(\mathit{mary})$, and so on, until the top is reached. Compositional interpretation schemes often go with bottom-up semantic construction, but it is important to note that this need not be the case. The equations in (10) satisfy the semantic compositionality requirement, but a solution can be found by traversing them in arbitrary order. In Muskens (1995) it is argued that this order-independence of interpretation is computationally useful. Note that the possibility of casting the semantics of (9) in the form of a system of equations that is solvable by means of a series of simple substitutions rests on the absence of variables. Had any of the intermediary representations contained a free variable, at least one substitution would

not have been possible. Muskens (1995) discusses a simple technique ('internalising the binding mechanism') which overcomes this difficulty and makes it possible to always represent semantic values by means of systems of equations.

While these considerations show that Compositionality and Bottom-up Construction should be separated at least on a conceptual level, the following quotation from Nerbonne (1995) introduces still another concept, namely that of the order in which terms are *evaluated*.

> Compositionality concerns the relation between syntax and semantics — i.e. (static) linguistic structure, not processing. Nonetheless, the compositional view often extends to a natural and popular interpretation, that of bottom-up processing. This is a natural interpretation because compositional semantics specifies the semantics of phrases via functions on the semantics of their daughters. It is natural to evaluate arguments before attempting functions to them (although partial and so-called 'lazy' evaluation schemes certainly exist).

Here the question is not so much in which order a final semantic representation of some expression is computed, but in which order the various parts of that representation are evaluated on a given model. Consider e.g. a disjunction $\varphi \vee \psi$. A strict bottom-up evaluation scheme would (a) compute the value of φ, (b) compute the value of ψ (not necessarily in this order), and then (c) take the maximum of the results. Lazy evaluation, on the other hand, would allow concluding that the value of $\varphi \vee \psi$ is 1 as soon as one of the disjuncts evaluates to 1. There is clearly a computational advantage to be gained here. The input-output behaviour of both evaluation schemes is the same only on the condition that the evaluation of a formula always returns a value. When this is not the case, for instance because information of a presuppositional character can render one subexpression undefined, empirical considerations may bear upon the question which evaluation scheme should be chosen.

Note that a radical interpretation of Bottom-up Evaluation would make it very unattractive from a computational point of view. In (1) we have given the value of (1c) as (1e), but this result was obtained after tacitly performing simplifications on the basis of λ-conversion. Evaluating (1e) on a finite model should be feasible if the model is not too large, although each quantifier essentially requires inspecting all elements of the domain. But Bottom-up Evaluation requires that daughters be evaluated before their mothers are, so it seems to require that the value of *every*, i.e. of $\lambda P' \lambda P \forall x[P'(x) \rightarrow P(x)]$, on the model given be computed before it is combined with the value of *student*, S,

and so on. As this needlessly gives a gigantic explosion of the number of computations involved, it must be concluded that Bottom-up Evaluation on a strict interpretation is extremely unattractive. Whether there can be more acceptable weaker interpretations of this evaluation scheme remains to be seen, but it is clear that Bottom-up Evaluation and Compositionality should not be confused.

A technique somewhat similar to lazy evaluation that does result in a compositional process has been applied for computing meanings as early as in 1984 in the TENDUM system (see Bunt et al., 1984), where quantificational ambiguities where treated by postponing disambiguation in a process that constructs the meaning of a noun phrase as a pair $\langle \mathsf{quant}', \mathsf{nom}' \rangle$ where quant' represents the semantics of a quantifying phrase and nom' represents the semantics of the head noun with any modifiers. Only when NP meanings are used to compute clause meanings are the quant' and nom' parts combined to form NP meanings of the usual kind. This process is somewhat similar to the use of Cooper storage but is more clearly compositional.

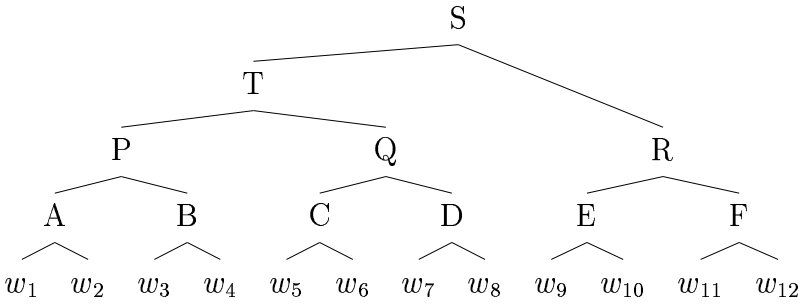
4. The ambiguity problem

The design of a process that computes meanings for a nontrivial range of sentences meets one obstacle more than any other: the overwhelming amount of ambiguity and vagueness inherent to natural language. Compositionality tells us that we can combine word meanings into the meanings of phrases, and phrase meanings into the meanings of sentences. But ambiguity and vagueness present problems to this at all points in the process. It presents problems at the lexical level, since most words are ambiguous (as well as vague), so for a start we have to assume a *range* of meanings for every word rather than a single meaning. Disregarding vagueness for the moment, we can hope that these ranges are finite sets, but this isn't entirely certain. Consider the meanings of prepositions like 'at', 'on', 'in'; these words allow an enormous range of semantically different uses, where it often seems rather arbitrary why one preposition should be used rather than another one (as witnessed by the difficulty of chosing the right prepositions in a foreign language). For many words, in particular for many of the most frequently used words, we have to assume large sets of meanings. The combination of words to form phrases, and of phrases to form sentences, gives rise to additional ambiguities: those caused by syntactic ambiguities, such as PP attachment, as well as those that have no syntactic

reflection, for instance when scope-bearing expressions are combined to form larger structures. Together these phenomena cause an explosion of the number of alternatives that must be considered by a process that is able to compute all the meanings of a nontrivial sentence. To appreciate the size of this problem, consider the following example.

Suppose (rather modestly!) that every word has 3 meanings. Take a simple sentence, like the present one, of twelve words length. Assume that the sentence is analysed in a strictly binary fashion, as illustrated in the tree diagram (11). For each of the nodes A, B, C, D, E and F there are 9 possible compound meanings to consider. At the next level up, for P, Q and R there are $9 \times 9 = 81$ choices. For node T, we have $81 \times 81 = 6.561$ possibilities, and at the top level we have $81 \times 6.561 = 531.441$ $(= 3^{12})$ possibilities. In other words, a modest degree of lexical ambiguity alone gives rise to half a million meanings for a simple sentence.

(11)



When we add syntactic and purely semantic (phrasal) ambiguity, we get truly astronomical numbers of interpretations. For instance, Hobbs and Shieber (1987) argue that the sentence

(12) Some representatives of every department in most companies saw a few samples of every product

has 42 valid alternative scopings.

More generally, in a sentence with five NPs, like (12), we have, depending on the constraints on scoping implied by the syntactic relations between the NPs, any number between 16 and 120 of alternative NP scopings. Many noun phrases also have a collective / distributive ambiguity, as illustrated by *"We lifted the piano"* vs. *"We ate an apple"*. Clearly, all these forms of ambiguity are independent, so we can have

sentences with many lexical ambiguities on top of scope ambiguities and ambiguities of the collective / distributive kind.

The third major source of ambiguity, that of alternative syntactic parses, is again in essence independent of the first two sources. And many other sources of ambiguity can be added, such as intensional versus extensional interpretation, literal versus figurative, metaphorical and idiomatic readings, the count/mass ambiguity (*"Mary had a little lamb"*), and restrictive versus presuppositional interpretation of relative clauses. These are just a few of the many other forms of ambiguity that are always on the lurch.

When we move to the level of discourse structure things get even worse. Not only are there new kinds of ambiguity to account for, as different attachments to discourse structure may be possible (see Gardent and Webber, 1998), but in addition a text consisting of n sentences, which are each m-ways ambiguous, will have m^n readings, as in the case of lexical ambiguities. In general, if a text grows, the number of its readings grows exponentially with it. This, essentially, is why the straightforward pipelined architecture implied in section 2 urgently needs amendation. The architecture essentially is a generate-and-test procedure, but since the number of readings to be generated is exponential in the length of the input, the algorithm is intractable and cannot be carried out, either by language users or by computers.

Clearly, the pervasive phenomenon of ambiguity in natural language makes it extremely difficult to design effective processes for computing the meanings of a nontrivial range of sentences. It is no wonder that the design of such processes has so far not been very succesful. Of course, not all the possibilities we have been considering in the above calculations are actually possible when the sentence is used in a certain context. The particular discourse domain will rule out many lexical meanings and combinations of those, and will also make certain scopings and quantifier interpretations very unlikely. What becomes especially clear, though, is that the number of possibilities to consider is incredibly large, and that a compositional process that would first calculate all logically possible disambiguated sentence meanings, and subsequently use context information to select the intended contextual utterance meaning from those, is not computationally viable.

At this point we may also note a strange aspect of the principle of compositionality that we did not yet consider. It speaks of "*the meaning* of a compound expression". The use of the singular *the meaning* is common in formulations of the compositionality principle, but clearly has no basis in reality: expressions in natural language hardly ever

have one single meaning. Speaking of *the meaning* is reasonable only when applied to *utterances*, where often only one of the many possible meanings of the sentence is contextually possible or relevant. This is why people can use language without constantly dealing with millions of possible meanings. (But as already noted, the meanings of utterances by their very nature do not obey Compositionality.)

5. Computing utterance meaning

5.1. *Sentence meaning and utterance meaning*

Research in computational semantics is concerned both with the algorithmics of sentence semantics and with the computation of utterance meaning, the ultimate dream being to be able to effectively compute the meanings of concrete manifestations of language, thereby allowing computers for instance to extract meaning from texts and to interact intelligently in dialogues with a human user.

Utterances, the concrete manifestations of sentences (or phrases, or other linguistic constructs), come with physical properties such as intonation, temporal structure and loudness in the case of a spoken utterance, and lay-out and punctuation marks in the case of a written utterance. These physical properties contribute to the meaning of an utterance. Moreover, an utterance appears in a certain context: it has a speaker, an addressee, a time and a place of occurrence in the case of a spoken utterance, and in the case of a textual utterance it has an author, a reader, and also a temporal and spatial context of occurrence. These and other contextual properties also contribute to the meaning that an utterance has.

For computing utterance meanings, we thus have three sources of information:

1. linguistic information: the semantic information carried by the component words and phrases and by the syntactic composition; anything that goes into the meanings of the sentence that is uttered;
2. physical utterance information: the information carried by the physical properties of the speech signal or the textual form of the utterance;
3. context information: the information that the speaker and hearers have about the domain of discourse; their (knowledge of their) respective goals and purposes; the spatio-temporal properties of

the situation in which the utterance occurs, etc. (see further Bunt, 1999).

Since sentences, in contrast with utterances, are theoretical constructs that have no existence outside linguistic analysis and discussion, the computation of sentence meanings is either of purely theoretical interest, or of potential interest for the computation of utterance meanings. Let us therefore consider the possible contribution of sentence meaning to the computation of utterance meanings.

5.2. *The contribution of sentence meaning to utterance meaning*

The computation of utterance meanings, as indicated above, involves combining linguistic and physical utterance information and context information, whereas computing sentence meanings only takes linguistic utterance information into account.[4] For an utterance u of a sentence S, the sentence meaning of S contributes directly to the computation of the meaning of u only when the processing of the *linguistic* information in u forms a separate stage. Assuming that physical utterance information and context information, being rather different types of information, are taken into account in separate stages as well, the straightforward organization of such a process would be as follows.

(13)
1. Compute the possible readings of S from the linguistic utterance information.
2. Apply physical utterance information to filter out certain unintended readings of u and to add pragmatic meaning aspects (see below).
3. Use context information to select the most plausible and relevant reading(s) of u.

Given the astronomical numbers of readings that we have seen to be involved in sentence meaning computation, this is not a computa-

[4] Note that the study of sentence meaning in formal semantics keeps context information out of the door by concentrating on isolated sentences, or by taking context into account in an extremely limited fashion, taking for instance a small amount of linguistic context (previous discourse) into consideration for dealing with anaphora, or taking parametric notions of speaker, hearer, time and place into account for the interpretation of indexicals, as in Montague (1968). But computational semantics must consider a much richter notion of context and its pervasive pervasive influence on meaning, which involves reasoning with speaker and hearer beliefs and with a full picture of interactive situations, in order to establish the intended meanings of words and phrases.

tionally feasible approach. To obtain a more efficient process, one idea is to take nonlinguistic information into account at an earlier stage, in order not to generate astronomically many readings in the first place. Concerning the use of physical utterance information, the following approaches can be found:

1. *'Prosody for pragmatics'*: prosodic information is considered as contributing to 'pragmatic' aspects of meaning (recognition of speech act type; topic-focus articulation, etc.). This is typically done after the processing of linguistic utterance information. In this case, physical utterance information is used to enrich the meanings generated from linguistic information, rather than to cut down the number of readings.
2. *'Punctuation for pragmatics'*: similarly for textual utterances, using e.g. end-of-sentence punctuation marks for speech act interpretation and intrasentential punctuation marks for information packaging analysis.
3. *'Preprocessing for punctuation'*: in a stage preceding the syntactic and semantic sentence analysis, punctuation marks are stripped from the utterance, and are interpreted perfunctorily.
4. *'Punctuation as linguistic information'*: punctuation marks are considered as linguistic elements and incorporated in the grammar. In this case (some of) the physical and linguistic utterance information are truly integrated.

Only in the latter case is physical utterance information used in a way that may lead to the generation of a smaller set of readings. This approach is found only rarely and in incomplete ways, since the incorporation of punctuation and layout into grammar formalisms is largely unchartered territory. Note that this approach does not have a separate phase of computing sentence meanings. The other approaches do have a separate phase of sentence meaning construction, and they do not make the organization (13) computationally less unattractive.

Concerning the early use of context information, one obvious possibility is to use knowledge of the domain of discourse to restrict the possible meanings of lexical items. For ambiguity at the phrasal level, a technique that is commonly used in language processing systems is to use domain information to rule out certain combinations of lexical meanings. This can be done by type checking, either interleaved with the construction of sentence meanings or as a filtering step afterwards. Although the use of domain information may help to reduce the number of readings, in general these approaches still result in the generation of

large sets of readings, most of which have to be discarded through much more sophisticated ways of using context information, since finding the intended meaning of an utterance is more a matter of taking into account what is contextually coherent, plausible and relevant than of deciding what is semantically possible.

To improve this situation, two approaches seem possible:

- Apply context information interleaved with linguistic information and block the generation of most sentence meanings at an early stage.
- Do not first generate alternative unambiguous possible meanings, but use the linguistic information to generate structures that capture *constraints* on the meanings that an utterance of the sentence may have. Use context information to add further constraints, thereby resolving ambiguities.

The first of these approaches is extremely difficult in practice, since the application of context information involves complex reasoning with pieces of information which during the linguistic information processing are only partly available. This poses problems both for the reasoning involved, and for the organization of interleaved linguistic information processing and reasoning. The second approach has been developed in computational semantics under the name of *underspecified semantic representations*. We consider this approach and its significance for effective utterance meaning computation in the next section.

6. Underspecified representations

Traditionally, semantic representations are expressions in a logical language that represent 'disambiguated' natural language sentences. An *underspecified* semantic representation of a sentence, or USR, is a formal representation of the semantic information expressed in the sentence, where anything that cannot be resolved on the basis of linguistic information is left unspecified. For example, an underspecified semantic representation may leave the relative scopes of quantifiers or the attachment of modifier phrases unspecified. An USR can thus be regarded as a shorthand for the representations of a number of unambiguous meanings, or alternatively as a specification of the constraints that all unambiguous meanings have to satisfy.

The idea of underspecified semantic representations is often traced back to the 'Quasi-Logical Form' (QLF), a particular form of underspec-

ified representation that was pioneered in the Core Language Engine system (Alshawi, 1992), especially for leavimg quantifier scopes unresolved. For example, the sentence *"Every students owns a book"* would have the QLF (14), which leaves the relative scope of the two quantifiers underspecified:

(14) `[own, qterm(<t=quant,n=sing,l=every>,X,[student,X]),`
`qterm(<t=quant,n=sing,l=a>,Y,[book,Y])]`

The use of QLF representations has been well publicized and has been very influential. However, the idea has a longer history. In the seventies the designers of the PHLIQA question answering system developed the idea of building semantic representations that would leave lexical ambiguity and vagueness unresolved by means of metavariables (see Medema et al., 1975; Bronnenberg et al., 1979; Bunt, 1981; Scha, 1983). In the TENDUM dialogue system, semantic representations were constructed leaving various kinds of structural ambiguities underspecified, such as collective/distributive readings, count/mass ambiguities, and modifier attachment ambiguities (see Bunt, 1985). Recently other, more sophisticated forms of underspecified representation have been defined in Discourse Representation Theory (Reyle, 1993), in the DenK project (Kievit, 1998) and in the Verbmobil project (Bos et al., 1994; Copestake et al, 1995).

Underspecified representations have great computational advantages, for two reasons. First, they allow purely linguistic information processing to generate a very small instead of a very large number of representations. Second, they open the possibility of a *monotone process* of utterance meaning computation where context information is used in a constructive way to add constraints to underspecified representations, as opposed to the generate-and-test approach where context information is used to discard large numbers of unambiguous representations.

The use of underspecified representations has advantages not only for dealing with ambiguity, but also for approaching the problem of semantic *vagueness*. Vagueness is distinguished from ambiguity in that the variations in meaning of a word cannot be described by a finite set of well-delineated alternative readings. Vagueness is therefore in fact even more troublesome for effective meaning computation than ambiguity. But once we allow underspecification in semantic representations, we can also allow predicates and other referential terms to be underspecified as to what precisely they refer to, and use contextual information to make this more specific by adding constraints on reference.

A related problem for which underspecified semantic representations may provide a solution, is that of the appropriate degree of precision in referential meaning. Whereas the traditional notion of 'disambiguated sentences' presupposes the possibility of total lack of ambiguity and vagueness, natural language expressions in fact may be considered unambiguous in one context, while requiring more fine-grained disambiguation or 'precisification' in another. In Artificial Intelligence, this issue has been addressed by introducing the notion of 'granularity' in modelling world knowledge (see Hobbs et al., 1987). Indeed, rather than assuming an absolute notion of 'unambiguity', it seems more appropriate to consider ambiguity and vagueness *relative to a certain context.* On this view there is no absolute definition of the set of meanings of a given sentence; meaning itself is a context-dependent notion.

While underspecification in semantic representations does seem to form a powerful concept, its theoretical status, its formal definition, and its possible use in reasoning are still the subject of study and discussion. Concerning their theoretical status, a central issue is whether underspecified representations are to be regarded as merely compact notations, useful as intermediate structures in a disambiguation processes, or whether they have a theoretical significance of their own. One reason to think that they may have a significance of their own is that people sometimes use ambiguous utterances in situations where it would be unnecessary, irrelevant, or even unintended to disambiguate. In such a situation, an underspecified semantic representation would seem to be an adequate description of the end result of a hearer's interpretation process. The following example illustrates this.

Since I have moved to another house a number of boxes with books and papers are still piling up in the hall, waiting to be carried upstairs to my study. In the morning, before going to the office I say to my two sons:

(15) If you carry these boxes upstairs today, I'll give you an ice cream.

When I come home in the evening, my sons confirm: *"We carried the boxes upstairs, dad!"* and so I buy them an ice cream.

The sentence uttered in (15) is multiply ambiguous due to the quantifiers. The boxes may be carried upstairs individually or collectively, both in the way the boys act, and in the way the boxes are acted upon. I, the speaker, don't care whether the boys act collectively or individually (or get the help of the boys next door...), nor do I care whether the boxes are carried one by one or in groups (or whether they

are unpacked, and repacked upstairs...). I certainly would not want the boys to spend effort on the disambiguation of my quantifiers, looking for an unambiguous interpretation they should try to satisfy. *There isn't any!* Instead, what I really meant to say was: "If you carry the boxes upstairs *in some way or other,* then I will give you an ice cream", and I would hope that the boys would act on the basis of understanding that 'ambiguous meaning'.

This example shows not only that people can use ambiguous utterances for succesful communication but also that people are able to reason with ambiguous information – which is largely the same thing. When I come home at night and conclude that the boys deserve an ice cream, I combine two ambiguous premises to derive a conclusion:

(16)
The boys have carried the boxes upstairs
If the boys have carried the boxes upstairs,
I will give them an ice cream

I will give them an ice cream

It thus seems possible to apply a rule like Modus Ponens to such ambiguous premises. Moreover, it is hard to imagine that this reasoning process makes use of the possible disambiguations of the premises, the more so since I did not intend the quantification in the *if*-clause in (16) to be disambiguated in the first place.

Whereas (16) suggests that direct inferences are possible with underspecified representations, other examples can be found that suggest the opposite. Questions concerning the interplay between inferencing and disambiguation, and concerning the rules for direct inferencing with ambiguous premises, form a hot issue in computational semantics today (see e.g. van Deemter, 1995; König and Reyle, 1996; Reyle, 1995; Jaspars, 1999).

The observation that it is at least sometimes possible to reason directly with underspecified representations supports the view that such representations have a meaning of their own. This view gets additional suppport from the application of underspecification to vagueness. Consider example (17), uttered by a speaker while giving the hearer a present.

(17) This is something Canadian.

The word *"Canadian"* can in general mean a lot of things, such as made in Canada, acquired in Canada, located in Canada, characteristic of

Canada, symbolizing Canada, controlled by the Canadian government, owned by a Canadian company, having its headquarters in Canada, etc. The situation in which (17) takes place of course rules out some of the possible more specific meanings of *"Canadian"*, but it still leaves many possibilities open – especially if the present is wrapped. Now it would not seem natural for the receiver of (17) to assign one of the possible more specific meanings of the word to *"Canadian"* in this utterance, especially as long as the present is not unwrapped; rather, the receiver may be assumed to assign (17) a vague meaning that may be paraphrased as *"This is something relating to Canada"*. As the present is unwrapped, the receiver may make his interpretation more specific. This process can be modelled by introducing a dummy predicate as a metavariable in an underspecified representation like (18), as suggested by Hobbs et al. (1993).

(18) $This\, x_1 : NN(x_1,\texttt{canada})$

At some stage the dummy predicate NN ('relating to') may be instantiated by a more specific relation supplied by the context. In this example one could imagine the metavariable NN to be replaced by the more specific predicate $Coming\text{-}from$; when the speaker provides additional information about the origin of the present, this may be further specied as $Produced\text{-}in$, which in turn may be further specified if additional context information becomes available as, for example, $Grown\text{-}in$, $Manufactured\text{-}in$, or $Designed\text{-}in$.

This example suggests that, while NN is a semantically vacuous predicate at one end of a scale of specificity, and other predicates such as $Coming\text{-}from$ and $Produced\text{-}in$ are somewhere along this scale in the direction of greater specificity, there would not seem to exist a predicate of greatest possible specificity. Absolute precision in meaning seems to be an illusion, as is also suggested by the phenomenon of 'granularity' that we came across, and so we may have to admit that meaning representations should be considered to *always* be underspecified to some degree...

7. About this book

In the chapter *On Semantic Underspecification*, **Pinkal** provides an overview of the motivations underlying the use of underspecified semantic representations, of what they may be taken to mean, of how they can

be used in reasoning, and of their theoretical status. He notes that USRs are motivated not just by their advantage in efficient natural language processing in the face of ambiguity, but also in allowing *robust* processing in the case of incomplete linguistic information, as often happens in speech understanding; moreover, in some cases it may be irrelevant or even undesirable to disambiguate an underspecified representation. Concerning the meaning of underspecified representations, Pinkal argues that the view of an USR as a partial description of logical formulae, or as a disjunction of its possible unambiguous instances, is inadequate, as is the view that an USR is semantically equivalent to the disjunction of the set of unambiguous logical formulae which it describes. In order to understand what an USR means, Pinkal examines various possibilities for defining entailment relations for USRs. He suggests that a cautious interpretation in terms of the possible readings of USRs seems most appropriate: an USR A entails an USR B iff every possible reading of B is classically entailed by every reading of A, but he also argues that the situation is complicated by phenomena of discourse parallelism. He therefore suggests a distinction between two entailment concepts, a 'dynamic' one that takes parallelism constraints into account, and a 'static' one that does not. In view of the interaction that may occur between discourse parallelism phenomena and the possible readings of an USR, Pinkal argues that underspecified representations form a layer of information which may be truth-conditionally irrelevant, but which is indispendable for discourse semantics.

Ramsay, in his chapter *Dynamic and Underspecified Interpretation without Dynamic or Underspecified Logic* addresses the demands on logical formalisms for underspecified meaning representation, as well as the demands that follow from the idea that utterance meanings should be viewed as context-changing operations. He argues, contra Groenendijk & Stokhof (1990, 1991) and many others, that there is no need for a special dynamic logic, and contra Reyle (1993) and many others that there is no need either to develop special logics for underspecification. He argues that very minor extensions to first-order logic are adequate both for accounting for the dynamics of meaning and for dealing with underspecification, by allowing belief sets of various kinds to be represented by propositions and using a constructive interpretation of the underlying logic.

Asher and Fernando are concerned with the utterance meaning disambiguation problem from the perspective of Discourse Representation Theory. In the chapter *Labeled Representations, Underspecification*

and Disambiguation they discuss the effects of discourse on disambiguating (sub)expressions and, treat the problem of computing those effects formally by imposing a labeling structure over discourse representation structures. This leads to barebones forms of segmented and underspecified DRT. They consider the possibility of marrying SDRT and UDRT, and argue that this can be done, with rather dramatic consequences for discourse interpretation. By allowing underspecification at all levels, they show that the requirements on contextual update seem to reduce, disambiguating only when the gains are worth the computational effort.

Richter and Sailer's chapter proposes a new approach to underspecified semantics in HPSG. They show that a general framework for underspecified semantic representation languages can be reconciled with the traditional logical architecture of HPSG as outlined in (Pollard and Sag, 1994), and as formalized in (King, 1994). As an example of a semantic object language that can be treated within this scenario, an extensional typed logic with lambda abstraction is extended to an underspecified representation language as described in (Bos, 1995), and the resulting language is modeled in HPSG.

Zadrozny tackles the problem of simplicity in grammars via the *minimum description length* principle of Rissanen (1982) and uses this principle to give more bite to the principle of compositionality. Compositionality in itself is empirically vacuous (Zadrozny, 1994) proves a result that he claims entails this vacuity). Informally researchers have always agreed that the functions composing the meanings of complex expressions out of the meanings of their parts should be simple, but up till now no formal characterisation of simplicity was forthcoming. Clearly, the problem of simplicity in linguistic descriptions is deep and important. Zadrozny's idea to use the minimum description length principle here seems to have much wider applications.

Van Genabith and Crouch in their chapter replace the static meaning representation language in the LFG linear logic based glue language semantics approach (Dalrymple et al., 1996; 1997) with a 'dynamic' representation language (Muskens, 1994b; 1996). This move extends the original approach to discourse phenomena and can be combined with an approach to underspecification developed in (Crouch & Genabith, 1996). On the other hand it provides linear logic based approaches to quantifier scope and underspecification for dynamic semantics. QLF and UDRT style interpretations are sketched for a set of

linear logic premises thus obtained. The results are briefly compared with some alternative approaches discussed in the literature.

Kyburg and Morreau in their chapter *Vague Utterances and Context Change* are concerned with the semantic representation of vague utterances. They present two context update functions that model the context change required to accomodate utterances containing adjective-noun combinations, like *fat pig* and *tall man*. In these models, the extensions of such expressions are sometimes stretched to suit the purposes of the participants in a dialogue: for example, a borderline fat pig may come to be called a fat pig following an utterance of *We will begin with the fat pig*. In the first model, the context change brought about by such an utterance is defined syntactically, in terms of the sentences representing the prior context. An updated context includes as much of the prior context as is consitent with the sentences conveyed by the utterance. This model is too permissive, though, and so an alternative, semantic model is suggested in which the updated context is instead defined in terms of changes to the *models* that support the context. On this view, an utterance can only bring about a new interpretation for a vague predicate according to which it is more precise.

Cooper's chapter *Using Situations to Reason about the Interpretation of Speech Events* applies the situation-theoretic notion of 'restriction' to account for information about an utterance conceived of as an event (a speech event) and for background information provided by the context. Also using role labels associated with abstracts to link various parts of an utterance with roles in the interpretation, he shows how a Montague-like compositional interpretation process can be obtained. Some problems are pointed out which are in part conceptual and in part technical, having to do with the computation of β-reduced λ-abstracts with restrictions. He then considers the possibilities of achieving the effect of restrictions and role labels with proof-theoretic tools which would allow to employ a simpler situation theory, and also possibly to use techniques similar to those used in type-logical approaches to grammar.

Kaplan and Schubert's chapter is concerned with modeling one of the most important aspects of the context of an utterance: the beliefs of the speaker, including his beliefs about the beliefs of the addressee. One of the problems in modeling beliefs is that of logical omniscience, which can sometimes be finessed in practice: when reasoning about another agent's belief the reasoner only has finite resources himself, so he will only discover some of the many conclusions that a conversational

partner supposedly believes according to the traditional possible-worlds models of modal logic. This finessing, called 'simulative inference', has been discovered by AI researchers and has been used since the early 1970s (Moore, 1973). Since agents discover only some of the consequences of their beliefs, as they have limited computational resources, a truthful model of beliefs should include a model of computation, like Konolige's deduction model of belief. Konolige's model allows some unrealistic agents and prohibits some realistic ones, therefore Kaplan and Schubert propose a different one, where the computational mechanism for deriving new beliefs from old ones can be any algorithm guaranteed to use finite amounts of time. Using this model, they characterize the conditions under which it is appropriate to reason about other agents by simulating their inference processes with one's own.

The chapter authored by **Meyer Viol, Kibble, Kempson and Gabbay** considers the use of Hilbert's ϵ-operator as a means to obtain the kind of underspecified representations of scope that are widely felt to be needed in computational semantics. The ϵ-calculus (there is also a dual universal τ-operator) is set up as a labelled deductive system here. Noun phrases are associated with certain 'prototerms,' but determination of the term dependencies in those prototerms is delayed by the parsing process. Terms project certain metavariables which can be instantiated later and these instantiations not only fix scope relations but also anaphoric dependencies. The approach treats noun phrases essentially as (type e) *terms* and the chapter analyses wide scope effects as falling together with anaphora.

In a somewhat related development **Schubert** observes that the so-called 'donkey' anaphora that have played a role in logic since the middle ages are all-pervasive in natural language and that much of the 'common-sense' knowledge needed for ordinary reasoning and language understanding characteristically involves them. Donkey anaphora have played a motivating role in the development of dynamic forms of semantics for natural language, but Schubert moves to a more standard logic by considering a procedure of *Dynamic Skolemization*. This procedure involves (i) introducing a Skolem constant for a given existential quantifier, as is also common in theorem proving, and (ii) stipulating a supplementary condition relating the existential statement under consideration with its skolemized form. The resulting representations are context-independent and admit a standard semantics, while referential connections can be made for instances of functional anaphora that are problematic for dynamic semantics.

Ginzburg in his chapter *Semantically-based Ellipsis Resolution with Syntactic Presuppositions* offers a variety of syntactic, semantic and processing reasons which suggest that the resolution process for 'short answers' in dialogue is problematic for existing approaches to ellipsis resolution. Short answers seem to maintain a limited amount of parallelism with their source: even an unbounded number of dialogue turns away, the short answer must bear the category assigned to the corresponding wh-phrase in the source. Ginzburg offers an account of the resolution process which also attempts to explain this unbounded syntactic dependency. He shows how an interrogative meaning can encode a specification for focus which includes constraints on structural realization. This involves developing the notion of a λ-abstract whose argument roles carry syntactic appropriateness restrictions. It is suggested that such a notion is already in essence presupposed in sign-based grammars such as HPSG, and that, with certain independently motivated modifications, such abstracts have similar semantic expressiveness to abstracts that lack such appropriateness restrictions.

Krahmer and Piwek address the question how Van der Sandt's theory of presupposition projection, which is widely acknowledged to give best coverage of the data, can be combined with a theory of *world knowledge*. The addition of some form of world knowledge to Van der Sandt's theory would obviously further improve its coverage and could make precise some explanations which are now intuitively appealing but informal. The way Krahmer and Piwek go about is to cast the theory, which was originally formulated in Discourse Representation Theory in a *Constructive Type Theory* of the Martin-Löf variety. This is argued to facilitate the interaction between world knowledge and presupposition projection. The approach is illustrated by considering cases of bridging and conditional presupposition.

Stone and Hardt's chapter concerns the interpretation of sloppy anaphora and starts with some intriguing examples that show that the phenomenon is really more general than the usual examples would suggest. Sloppy anaphora involve some constituent of category XP containing a 'sloppy variable' of category YP controlled by an element C1. The interpretation of C1 contributes to the meaning of XP via YP. When an anaphoric element XP$'$ further in the sentence refers to XP, the role of C1 may have been taken over by a new controller C2. The usual instantiations for XP are NP and VP and commonly only the possibility of NPs is considered for sloppy variables, but Stone and Hardt make it plausible that XP and YP can almost freely be chosen

from NP, VP, modality and tense. They obtain an interpretation of sloppy anaphora using a dynamic semantics which allows the storing of dynamic objects. Possible context changes between XP and XP′ then account for the possible change in interpretation. Thus a uniform treatment of sloppy identities in various categories is obtained.

In their chapter *Linking Theory and Lexical Ambiguity: the Case of Italian Motion Verbs* **Dini and Di Tomaso** describe an approach to the treatment of locative expressions in romance languages which eliminates one of the possible sources of inefficiency of practical NLP systems, viz. presuppositional ambiguity. They show that, contrary to what is often believed, romance locatives do not have two distinct senses, a static one (location) and a dynamic one (goal). On the contrary, only the static sense needs to be assumed, while the goal interpretation is achieved as the result of the interaction of three independently motivated modules of the grammatical organization, viz. the theory of Aktionsart, Linking Theory, and Control Theory.

Last but not least, **Reinhard's** contribution *A Disambiguation Approach for German Compunds with Deverbal Heads* considers German noun-noun compounds such as *"Mitarbeiterbesprechung"*, *"Kinobegeisterung*, and *"Jugendgefährdung"*. Such compounds are not only extremely frequent in German, they are also ambiguous between various readings. Reinhard focuses on noun-noun compounds with a deverbal head, which have the property that the first constituent either satisfies an argument role of the second or stands in a more or less specifiable modifying relation to it. The goal here is automatic prediction of the correct relationship, so that the meaning of the compound can be computed on the basis of the meanings of its parts. The research was carried out on the basis of a corpus (the *Frankfurter Rundschau*) and a number of generalisations about the argument inheritance behaviour of certain nouns could be made. This gives default interpretations in many cases and the analysis also rules out certain readings. Interestingly, the empirical work done here also refutes some theoretical stipulations about argument inheritance that were made in previous literature.

References

Alshawi, H. (1992) *The Core Language Engine.* Cambridge, MA: MIT Press.

Alshawi, H. and R. Crouch (1992) Monotonic semantic interpretation. In *Proceedings 30th Annual Meeting of the Association for Computational Linguistics*, 32–38.

Asher, N. (1993) *Reference to Abstract Objects in Discourse.* Dordrecht: Kluwer.

Bos, J. (1995) Predicate Logic Unplugged. In *Proceedings 10th Amsterdam Colloquium.*

Bos, J.; E. Mastenbroek, S. McGlashan, S. Millies, and M. Pinkal (1994) A compositional DRS-based formalism for NLP-applications. In H. Bunt, R. Muskens and G. Rentier (eds.) *Proceedings International Workshop on Computational Semantics*, Tilburg.

Bronnenberg, W.J.; H.C. Bunt, J. Landsbergen, P. Medema, R. Scha, W.J. Schoenmakers, and E. van Utteren (1979) The question-answering system PHLIQA 1. In L. Bolc (ed.) *Natural communication with computers.* London: MacMillan.

Bunt, H.C. (1981) *The formal semantics of mass terms.* Ph.D. dissertation, University of Amsterdam.

Bunt, H.C. (1985) *Mass terms and model-theoretic semantics.* Cambridge, UK: Cambridge University Press.

Bunt, H.C. (1999) Iterative context specification and dialogue analysis. In: H.C. Bunt and W.J. Black (eds.) *Abduction, Belief and Context in Dialogue. Studies in Computational Pragmatics.* Amsterdam: John Benjamins.

Bunt, H.C. and A. van Horck (eds.) (1996) *Discontinuous Constituency.* Berlin: Mouton de Gruyter.

Bunt, H.C.; R.J. Beun, F. Dols, J. van der Linden, and G.O. thoe Schwartzenberg (1984) The TENDUM dialogue system and its theoretical basis. *IPO Annual Progress Report* 19, 107–112.

Cooper, R. (1983) *Quantification and Syntactic Theory.* Dordrecht: Reidel.

Copestake, A.; D. Flickinger, and I.A. Sag (1995) Minimal Recursion Semantics, An Introduction. Ms. CSLI, Stanford Unviersity. `coan.stanford.edu/ergo/parser.html`

Crouch, R. and J. van Genabith (1996) Context change and underspecification in glue language semantics. In Butt, M. and King, T.H, (eds.) *Proceedings of the First LFG Conference*, RANK Xerox Research Center, Grenoble, France, 133–147.

Dalrymple, M.; J. Lamping, F. Pereira, and V. Saraswat (1996) A deductive account of quantification in LFG. In M. Kanazawa, C. Pinon, and H. de Swart (eds.) *Quantifiers, Deduction and Context.* CSLI Publications, No. 57, 33–57.

Dalrymple, M.; J. Lamping, F. Pereira, and V. Saraswat (1997) Quantifiers, anaphora, and intensionality. *Journal of Logic, Language, and Information* 6(3), 219–273.

Declerck, T (1996) Modelling information passing with the LFG workbench. In Butt, M. and T.H. King (eds.) *Proceedings of the First LFG Conference*, RANK Xerox Research Center, Grenoble, France.

Deemter, K. van (1995) Towards a logic of ambiguous expressions. In K. van Deemter and S. Peters (eds.) *Semantic Ambiguity and Underspecification.* Stanford: CSLI Publications, 203–238.

Gardent, C. and B. Webber (1998) Describing discourse semantics. In prep.

Groenendijk, J. and M. Stokhof (1990) Dynamic montague grammar. In Kalman, L. and L. Polos (eds.) *Papers from the Second Symposium on Logic and Language.* Akademiai Kiadoo, Budapest, 3–48.

Groenendijk, J. and M. Stokhof (1991) Dynamic predicate logic. *Linguistics and Philosophy* 14, 39–100.

Hobbs, J.R.; W. Croft, T. Davies, and K. Laws (1987) Commonsense metaphysics and lexical semantics. *Computational Linguistics* 13 (3-4), 241–250.

Hobbs, J.R.; M.E. Stickel, D.E. Appelt, and P. Martin (1993) Interpretation as abduction. *Artificial Intelligence* 63, 69–142.

Hobbs, J.R. and S.M. Shieber (1987) An Algorithm for Generating Quantifier Scopings. *Computational Linguistics* 13 (1-2), 47–63.

Janssen, T.M.V. (1997) Compositionality. In J. van Benthem and A. ter Meulen (eds.) *Handbook of logic and language.* Amsterdam: Elsevier, 417–473.

Jaspars, J.O.M. (1999) Structural Logics for Reasoning with Underspecified Representations. In *Proceedings First International Workshop on Inference in Computational Semantics,* Amsterdam, 69–82

Keller, W. (1996) Nested Cooper storage: The proper reatment of quantification in ordinary noun phrases. In U. Reyle and C. Rohrer (eds.) *Natural Language Parsing amd Linguistic Theory.* Dordrecht: Reidel, 432–437.

Kievit, L.A. (1998) *Context-driven natural language interpretation.* Ph.D. Dissertation, Tilburg University.

King, P.J. (1994) An Expanded Logical Formalism for Head-Driven Phrase Structure Grammar. Arbeitspapiere des SFB 340 59. Universität Tübingen.

König, E. and U. Reyle (1996) A general reasoning scheme for underspecified representations. In Ohlbach, H.-J. and U. Reyle (eds.) *Logic and its Applications. Festschrift for Dov Gabbay.* Dordrecht: Kluwer.

Konolige, K. (1986) *A Deduction Model of Belief.* San Mateo: Morgan Kaufmann.

May, R. (1977) *The grammar of quantification.* Ph.D. dissertation, MIT, 247–270.

Medema, P.; W.J. Bronnenberg, H.C. Bunt, J. Landsbergen, R. Scha, W.J. Schoenmakers, and E. van Utteren (1975) PHLIQA 1: multilevel semantics in question answering. *AJCL Microfiche 32.*

Montague, R. (1968) Pragmatics. Reprinted in Thomason, R, (ed.) (1974) *Formal Philosophy, selected papers of Richard Montague.* Yale University Press, 95–118.

Montague, R. (1973) The proper treatment of quantification in ordinary English. Reprinted in Thomason, R, (ed.) (1974) *Formal Philosophy, selected papers of Richard Montague.* Yale University Press, 247–270.

Muskens, R. (1994) A compositional discourse representation theory. In Dekker, P. and M. Stokhof (eds.) *Proceedings 9th Amsterdam Colloquium.* ILLC, Amsterdam, 467–486.

Muskens, R. (1995) Order-independence and underspecification. In *Dyana-2 Deliverable R2.2.C Ellipsis, Underspecification, Events and More in Dynamic Semantics.* ILLC, University of Amsterdam.

Muskens, R. (1996) Combining montague semantics and discourse representation theory. *Linguistics and Philosophy* 19, 143–186.

Moore, R.C. (1973) D-script: A computational theory of descriptions. In *Proc. 3rd International Joint Conference on Artificial intelligence,* 223–229.

Nerbonne, J. (1995) Compositional Semantics - Linguistics and Processing. In S. Lappin (ed.) *Handbook of Contemporary Semantic Theory.* London: Blackwell, 459–482.

Partee, B., A. ter Meulen and R. Wall. (1990) *Mathematical methods in linguistics.* Dordrecht: Kluwer.

Pinkal, M. (1996) Radical underspecification. In Dekker, P. and M. Stokhof (eds.) *Proceedings of the Tenth Amsterdam Colloquium.* ILLC, University of Amsterdam.

Pollard, C., and I. Sag. (1994) *Head-Driven Phrase Structure Grammar.* University of Chicago Press.

Reyle, U. (1993) Dealing with ambiguities by underspecification: Construction, representation and deduction. *Journal of Semantics* 10, 123–179.

Reyle, U. (1995) On reasoning with ambiguities. In *Proc. Seventh Conference of the European Chapter of the Association for Computational Linguistics,* 1–8.

Scha, R.J.H. (1983) *Logical foundations for question answering.* Ph.D. dissertation, University of Groningen.

Rissanen, J. (1983) A universal prior for integers and estimation by minimum description length. *Annals of Statistics* 11, 416–431.

Zadrozny, W. (1994) From compositional to systematic semantics. *Linguistics and Philosophy* 17 (4), 329–342.