Machine Translation **11:** 75–107, 1996.

Lexical Paraphrases in Multilingual Sentence Generation

MANFRED STEDE
Technische Universität Berlin, FB 13, Sekr. 5–12, Franklinstraße 28/29, 10587 Berlin, Germany

Abstract. We describe the architecture of a sentence generation module that maps a language-neutral "deep" representation to a language-specific sentence-semantic specification, which is then handed over to a conventional front-end generator. *Lexicalization* is seen as the main task in the mapping step, and we specifically examine the role of verb semantics in the process. By separating the various kinds of knowledge involved, for related languages (such as English and German) the task of multilingual sentence generation can be treated as a variant of the monolingual paraphrasing problem.

Key words: lexical semantics, paraphrases, multilingual generation

1. Introduction

1.1. Multilingual Sentence Generation

Curiously, research in natural language generation (NLG) has for a long time been content with producing text in a single target language, whereas the idea of multilingual generation (MLG) has gained popularity only in the past few years. This is surprising because NLG is concerned with mapping some "deep" representation (of whichever kind) to a linguistic one – and once that effort is undertaken, it makes sense to provide the mapping to multiple target languages, which opens up a range of new application possibilities.

The task of MLG is comparable to that of the "second half" of interlingual machine translation (MT), in which the interlingua representation that was built up from parsing the source text gets mapped to sentences of the target language. However, the status of an interlingua in MT is not quite the same as that of an MLG input representation; while an interlingua for MT purposes can be designed specifically around the relevant linguistic phenomena of the languages involved (see, for example, Dorr, 1993), the input to an MLG system is typically dependent on the underlying application. Specifically, as our approach was developed in a scenario of generating maintenance instructions for technical products, we take the language-neutral level to be *instantiated domain knowledge*. And the ontological decisions made in a domain knowledge base can differ from those amenable to generating language: depending on the application, the representation can be geared towards automated reasoning, simulation, etc., and the category distinctions need not always correspond nicely to the lexical categories of one or all of the target languages (see Stede and Grote, 1995, for discussion).

Nonetheless, the issues addressed in this paper are relevant for MT as well, because translation is often not possible without knowledge and inference capabilities; the problem of relating lexical meaning to background knowledge thus arises in MLG and MT alike.

Basically, any MLG system maps a language-neutral to a language-specific representation at some point in the overall process.[1] And this mapping can involve substantial re-structuring of representations,[2] either because of a mismatch between the categories needed for reasoning and those needed for generation, or because the individual target languages differ in terms of the sentence structure they use to verbalize an event.

1.2. PARAPHRASES

For language generation to be an interesting endeavor, the relationship between the language-neutral representation and linguistic output needs to be more sophisticated than a one-to-one correspondence. Only when the generator has a choice between different ways of saying roughly the same thing, questions of tailoring text to different audiences or situations can be explored. Generation becomes a matter of choosing the most suitable utterance from a set of paraphrases.

In the system presented here, the primary responsibility for mediating between language-neutral and language-specific representation rests upon the lexicon of the target language. Accordingly, we focus our attention on lexical differences between languages and on lexical variation within a single language. And specifically, we are interested in different verbalizations of *events* and therefore in verb semantics. For example, the event of someone uncorking a wine bottle can be described with the phrases *open the bottle*, *uncork the bottle*, or *remove the cork from the bottle*. The first characterizes the result of the event, whereas the others give more specific information on the manner of the action. The second phrase is essentially a shorthand for the third – *uncork x* means the same as *remove the cork from x*, and the two are said to differ in terms of *incorporation*. Depending on the intended focus of the utterance, any version can be the most appropriate. Imagine that we wish to add to the sentence the information that the cork was mouldy; this leaves only the option of using the *remove* phrase: *remove the moldy cork from the bottle.*

Other paraphrases exhibit predominantly *stylistic* variation, such as the difference between *They ordered me to go to Texas* and *They asked me to go to Texas*. Of course, stylistic variants need not always be so neatly parallel – they can involve different sentence structures, as in *Wayne considered leaving the house/Wayne entertained the thought of vacating the premises*.

In a multilingual environment, such structural differences can become a matter not of choice but of necessity. In automobile manual corpus studies (Rösner and Stede, 1994), for instance, we found the English instruction *Twist the cap until it* *stops* translated to German as *Drehen Sie den Deckel bis zum Anschlag* (lit. ‘Turn the cap up to the stopping point’): two clauses in English versus one clause with prepositional phrase in German – and indeed no “closer” translation is possible. In a different example, from the section on changing spark plugs: *Disconnect the wire from the plug*/*Ziehen Sie das Zündkabel von der Zündkerze ab.* (‘Pull the wire off the spark plug’), the English sentence expresses the resulting state of the action (wire and plug disconnected), whereas the German verb characterizes the physical activity bringing about the new state.

Research in contrastive linguistics (e.g. Hawkins, 1986, p. 28) has pointed out a general tendency for English to prefer more abstract verbs, while German more often uses specific or “concrete” verbs describing the manner of the action. One instance of this tendency is the case of *remove*, which in the automobile manual corresponds to numerous more specific German verbs that add information about the nature of the process and the topological relationship between the objects involved.[3]

Examples like these demonstrate the need for an explicit role of lexical semantics in MLG; however, such questions have so far received little attention in generation research. To overcome this deficit, we will create the possibility of finding a preferred paraphrase among a set of candidates, by developing structured lexical entries rich in information. The actual choice among paraphrases is not discussed here, though; instead, we limit the present task to making a range of verbalization alternatives available to a generator. Importantly, our system is designed in such a way that the monolingual paraphrasing task is straightforwardly extended to that of multilingual sentence generation – with the limitation that we are dealing only with languages that are quite closely related, specifically with English and German.

1.3. OVERVIEW OF THE PAPER

The paper is organized as follows. Section 2 describes the architecture of MOOSE, a module for multilingual sentence generation. The two levels of representation and the role of the lexicon in mediating between them are explained. Section 3 looks in more detail at the handling of verb meaning, which is the central device in mapping between the representations. The procedure accomplishing this mapping is explained in Section 4. Then, Section 5 describes the range of monolingual and multilingual paraphrases produced by the system and gives some detailed examples. Finally, Section 6 summarizes the work and draws comparisons to related research.

2. Two-Step Sentence Generation

The MOOSE sentence generator grew out of experiences with building the TECHDOC system (Rösner and Stede, 1994), which produces user instructions for technical products in multiple languages from a common representation. One initial assumption in designing TECHDOC posited that semantic sentence specifications be identical for the target languages English, German, and French; while this simplification is often tenable in the particular application domain, it is hard to defend in general. In response, MOOSE is designed as a sentence generation module that pays attention to language-specific lexical idiosyncrasies, and that can be incorporated into a larger-scale text generator (such as TECHDOC). A full description of MOOSE is given in (Stede, 1996b).

2.1. THE ARCHITECTURE OF MOOSE

Figure 1 shows an overview of the MOOSE system. We give a brief overall description here, and more details will be provided in the sections to come.

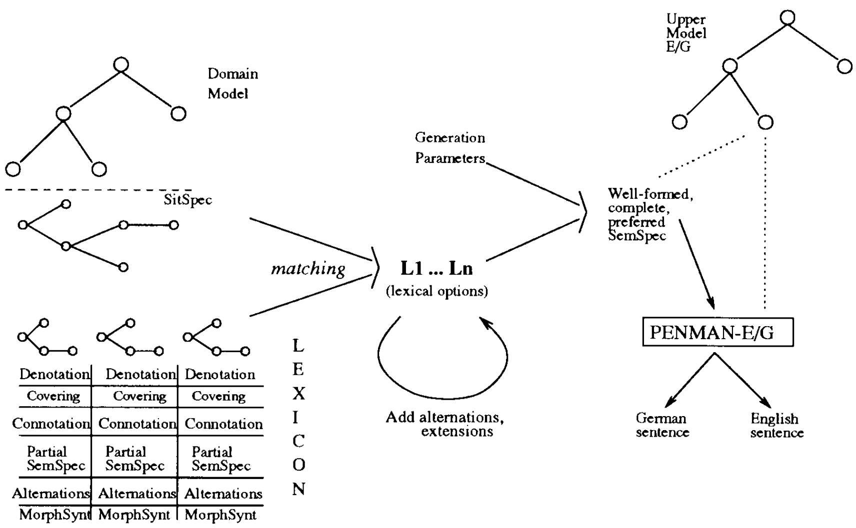

Figure 1. MOOSE system architecture.

MOOSE is built on top of the Penman front-end generator (Penman, 1989), to which various extensions were made, and a German variant that was developed for the TECHDOC domain at the Research Center for Applied Knowledge Processing (FAW), Ulm. The knowledge sources involved in the generation process are:

- the language-neutral domain model,
- the language-specific upper models,
- the language-specific generation grammars (implemented as systemic networks (Matthiessen and Bateman, 1991)),
- the language-specific lexicons.

The domain model (DM) is a taxonomy of concepts and relations holding the knowledge needed to represent entities of the domain, and to perform certain reasoning operations with them. These operations are independent of the language generation task and involve automatic planning and simulation of complex events;

accordingly, the basic ontological distinctions made in the DM are often not in simple correspondence with linguistic categories. The DM is implemented in the KL-ONE language LOOM (MacGregor and Bates, 1987).

An upper model (UM) (Bateman et al., 1990), on the other hand, is a taxonomy dedicated to the needs of language generation. The model represents the semantic distinctions a language makes and aids the generation grammar in producing the syntactic reflections of these differences. Roughly speaking, any difference in form corresponds to different upper model concepts. The central sub-hierarchy of the UM is a taxonomy of *process* types, to which configurations of participants and circumstances (see Section 3.1) are associated, which the Penman generator uses to derive appropriate verbalization patterns.

Whether, or to what extent, upper models are in fact specific to a single language or can be shared across languages, is an area of active research; see, for example, Bateman et al. (1991), and Henschel (1993). In MOOSE, we use the Penman upper model for English, with several modifications made in the TECHDOC project, and a German version, which was also developed for TECHDOC. The two taxonomies overlap to a good extent and are implemented with shared representations.

In the original Penman idea, the domain model is to be subsumed by the linguistic upper model (any DM concept must have a superordinate in the UM), so that Penman can infer the possible linguistic realizations for instantiated domain representations straightforwardly. TECHDOC also followed this principle. However, this method seriously limits the range of possible verbalizations, and in fact presupposes that the DM be structured along the linguistic categories of English. Furthermore, in a multilingual system of this architecture, the input to Penman must have the same structure for each target language; this is often tenable for languages that are closely related, but in general is too strict an assumption. To overcome this limitation, the MOOSE architecture separates the DM from the UM and treats the two as completely separate hierarchies. Accordingly, an explicit mapping step between the pre-linguistic and the linguistic representation is performed in order to achieve a wider range of paraphrasing possibilities. If an event concept in the DM were directly subsumed by a UM process type, no verbalization variants differing in terms of the process–participant structures could be produced. Therefore, instead of establishing direct subsumption links between UM and DM, MOOSE provides the connection between the two models indirectly, via the language-specific lexicon. As we will describe below, lexicon entries pair templates of DM representations with templates of sentence-semantic specifications.

The generator starts from a language-neutral *SitSpec*, a network of instantiated DM concepts, which could, for example, be the output of a simulation or planning module of the overall system. Generation then proceeds as follows.

1. The SitSpec is first matched against the denotations that are part of lexical entries (see Section 2.3), in order to determine the range of candidate lexemes for verbalizing the SitSpec. The result is a set of *verbalization options* (VOs):

lexical entries that are now associated with the information as to what parts of the input SitSpec they can express or *cover* (which is always a subset of or identical with that part of the SitSpec that matches the denotation of the lexeme).

2. For VOs that represent verbs, the applicable *alternations* and *extensions* of that verb are computed by a number of lexical rules, and the results are added to the set of VOs. This step will not be discussed in this paper (but see Stede, 1996a). In brief, the idea is the following. The initial matching phase considers only a minimal base form of a verb; hence, only that form has an entry in the lexicon, and more complex configurations are derived from it by the rules. For example, the lexical entry of a verb like *drain* describes only the configuration corresponding to a sentence like *The water drained from the tank*, while the locative alternation and the causative reading are derived by productive rules, leading to sentences like *The tank drained of the water; Tom drained the water from the tank; Tom drained the tank of the water.*
3. In the following step, a subset of VOs is determined from which a language-specific semantic sentence specification *SemSpec* is constructed, whose well-formedness is guarded by the language-specific UM. In addition to fulfilling the well-formedness constraint, the lexemes participating in SemSpec construction are chosen in accordance with generation parameters pertaining to brevity and stylistic features. The mapping procedure will be explained in Section 4, and an example illustrating its function will be given there. The topic of finding the preferred verbalization is not dealt with in this paper. In a nutshell, the features used for stylistic comparison in MOOSE are similar to those proposed by Hovy (1988); more details on our approach can be found in DiMarco et al. 1993.
4. Finally, the SemSpec is handed over to the surface generator, i.e. either to Penman or to its German counterpart. Both make use of the respective UM; the two UMs do in fact overlap to a large extent, as do the generation grammars, which are implemented with shared representations of system networks.

The idea of using the lexicon for mapping between different levels of representation in NLG is not new; a similar mechanism is used for instance in DIOGENES (Nirenburg and Nirenburg, 1988). However, MOOSE is the first system generalizing this step to a multilingual environment and performing the mapping in a principled manner that is grounded in an explicit treatment of lexical semantics, which previous systems have largely neglected.

2.2. Levels of Representation

A central assumption made in the design of MOOSE is that the “deepest” level of representation is in general not a linguistic representation; instead, an explicit transition between instantiated domain knowledge and a language-specific semantic sentence representation is seen as the central step in generation. The lexicon of the target language is instrumental in performing this transition.

2.2.1. *SitSpec*

The development of the domain model and the underlying ontology of MOOSE focused on the treatment of *events* so that they can be appropriately verbalized in different languages. The hierarchy of SITUATIONS, shown in Figure 2, is organized along a variant of the ontological categories proposed by Vendler (1967) and developed further, *inter alia*, by Bach (1986). We briefly discuss the three types of situation in our system in turn.

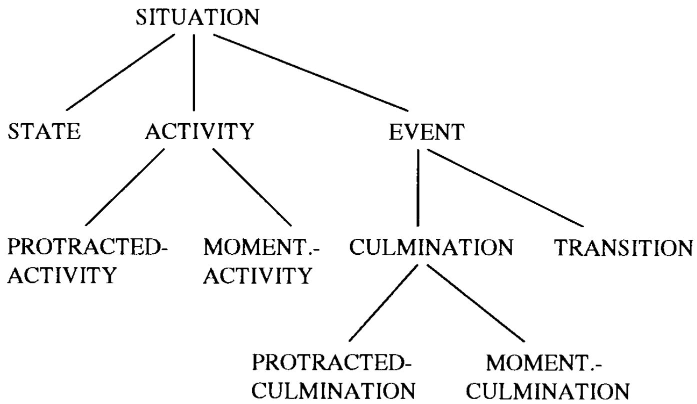

Figure 2. Situation types in the ontology of MOOSE.

STATES are seen much in the same way as Bach sees them: something is attributed to an object for some period of time, and the object is not perceived as "doing" anything. *The bottle is empty* is true for the bottle without it doing anything about it. We do not make further distinctions among states here.

ACTIVITIES were called "processes" by Bach, but we need this term on a different level of description (the UM). They are quite similar to states, but there is always something "going on": the water in the lake being calm is a state, but the water in the river flowing towards the sea is an activity. This case demonstrates that we see activities as independent of *volition*. We distinguish two subtypes here: PROTRACTED ACTIVITIES take place over an extended period of time, whereas MOMENTANEOUS ACTIVITIES occur in an instant. A linguistic test to distinguish the two is the "point adverbial": *Jill knocked at noon*. With a protracted activity, such a sentence is grammatically well-formed, too: *Jill slept at noon*. But, this sentence does not entail that Jill did not sleep immediately before and immediately after noon. Also, notice that the standard diagnosis for protracted activities, adding a frame adverbial such as *for an hour*, always produces an *iterative* reading when applied to a momentaneous activity: *to knock for an hour* does not mean that a single knock lasted that long, but that the activity is performed repetitively.

EVENTS are occurrences that have a structure to them; in particular, their result, or their coming to an end is included in them: *to destroy a building, to write a book*.

As their central feature we take them to always involve some change of state: the building loses its integrity, the book comes into existence, or gets finished. While Bach (1986) did not investigate the internal structure of events, others suggested that this needs to be done (e.g. Moens and Steedman 1988; Parsons 1990). Pustejovsky (1991a) treated Vendlerian accomplishments and achievements as transitions from a state Q(y) to NOT-Q(y), and suggested that accomplishments in addition have an intrinsic agent performing an activity that brings about the state change.

We follow this line, but modify it in some ways. Basically, we see any EVENT as involving a state change; an activity responsible for the change can *optionally* be present. A plain TRANSITION is necessarily momentaneous (*The room lit up*), whereas a transition-with-activity inherits its protracted/momentaneous feature from the embedded activity. We call these tripartite events CULMINATIONS. They are composed of a pre-state (holding before the event commences), a post-state (holding when the event is over), and an optional activity that brings the transition about. Generalizing from Pustejovsky's proposal, we take state transitions to be more than merely oppositions of Q(y) and NOT-Q(y); they can also amount to a gradual change on some scale, or involve other values. Also in contrast to Pustejovsky, we do not regard the presence of a volitional agent as responsible for any of the category distinctions; rather, the agentivity feature cuts across the categories discussed. Other aspects of the MOOSE ontology are designed following proposals by Jackendoff (1990), in particular his analysis of movement events.

Subsumed by the general ontological system, a domain model is defined that holds the concepts relevant for representing situations in a technical sample domain and that specifies the exact conditions for the well-formedness of situations. A *SitSpec* is thus a network of instances of DM concepts, and its root node is of type SITUATION. As an example, the EVENT of a person named Jill filling a tank with water is shown in Figure 3 in a graphical KL-ONE notation, with relation names appearing in boxes. The event combines the activity of Jill pouring water into the tank with the fill-state of the tank changing to full. A verbalization of this event can emphasize either of these aspects.

2.2.2. *SemSpec*

The level of SemSpecs is a subset of the input representation language that was developed for Penman, the sentence plan language (SPL) (Kasper, 1989). Therefore, a SemSpec, like an SPL expression, represents the configuration of process, participants, and circumstances to be expressed in the sentence, and abstracts from aspects of syntactic realization such as preposition choice or linear order of constituents. These decisions are made by Penman on the basis of the upper model types of the entities involved.

As opposed to a general Penman–SPL expression, though, a SemSpec may contain only UM concepts and no domain concepts; recall that the DM in MOOSE is deliberately *not* subsumed by the UM. Furthermore, since our system takes

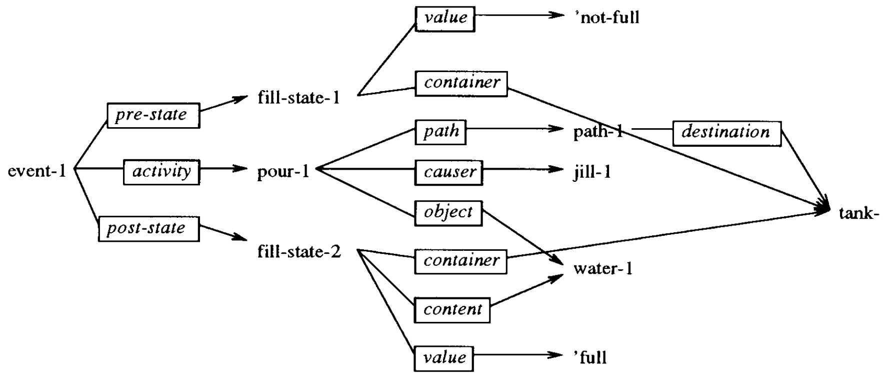

Figure 3. SitSpec representing a fill-event.

lexicalization as the decisive task in mapping a SitSpec to a SemSpec, the UM concepts referred to in a SemSpec must be annotated with `:lex` expressions; thus, a SemSpec is a lexicalized structure. Accordingly, we see the UM as a taxonomy of lexical classes.

A SemSpec is constructed from a SitSpec by selecting a process and mapping the SitSpec elements to participant roles or to circumstantial roles of that process, so that all elements of the SitSpec are covered. The concepts in the UM specify the conditions for well-formedness of SemSpecs by listing for every process the obligatory and optional participants and, sometimes, broad selectional restrictions on the fillers. To illustrate, Figure 4 gives four SemSpecs that MOOSE produces, amongst others, from the SitSpec shown in Figure 3. Sentences (1) and (2) express the activity of pouring in the main clause and add a verbalization of the post-state using the connective *until/bis*, which is chosen because the concept POUR is a subtype of PROTRACTED-ACTIVITY in the situation ontology (it can be performed for an extended period of time). Sentences (3) and (4), on the other hand, verbalize the achievement of the post-state and leave the precise nature of the activity aside. Notice that German uses the same lexeme *füllen* in (2) and (4), because the verb undergoes the locative alternation; the corresponding English *fill* does not, and therefore MOOSE needs to make a different lexical choice in (1): *pour*.

2.3. THE ROLE OF THE LEXICON

MOOSE is designed with the goal of strong lexical paraphrasing capabilities in mind. Therefore, its lexicon is rich in information so that lexical choices can be made on the basis of various generation parameters. A lexical entry in MOOSE has the following components:

Denotation. A partial SitSpec that defines the *applicability condition* of the lexeme: if its denotation subsumes some part of the input SitSpec, then (and only then) it is a candidate lexical option for the verbalization.

(1) *Jill poured water into the tank until it was filled.*

```
(x1 / anterior-extremal
   :domain (x2 / directed-action :lex pour_el
           :actor (x3 / person :name jill)
           :actee (x4 / substance :lex water_el)
           :destination (x5 / three-d-location :lex tank_el))
   :range (x6 / nondirected-action :lex fill_el
           :actee x5))
```

(2) *Jill füllte Wasser in den Tank, bis er voll war.*

```
(x1 / anterior-extremal
   :domain (x2 / directed-action :lex fuellen_gl
           :actor (x3 / person :name jill)
           :actee (x4 / substance :lex wasser_gl)
           :destination (x5 / three-d-location :lex tank_gl))
   :range (x6 / property-ascription
           :domain x5
           :range (x7 / quality :lex voll_gl)))
```

(3) *Jill filled the tank with water.*

(4) *Jill füllte den Tank mit Wasser.*

```
(x1 / directed-action :lex fill_el
   :actor (x2 / person :name jill)
   :actee (x3 / object :lex tank_el)
   :inclusive (x4 / substance :lex water_el))
```

Figure 4. SemSpecs and corresponding sentences.

Covering. The subset of the denotation nodes that are actually expressed by the lexeme. One of the constraints for sentence production is that every node be covered by some lexeme.

Partial SemSpec (PSemSpec). The contribution that the lexeme can make to a sentence SemSpec. By means of shared variables, the partial SemSpec is linked to the denotation.

Connotations. Stylistic features pertaining to formality, floridity, etc. (not discussed here; see DiMarco et al., 1993).

Salience assignment (for verbs only). A specification of the different degrees of prominence that the verb assigns to the participants (not discussed here).

Alternation rules (for verbs only). Pointers to lexical rules that compute the alternations applicable to the verb (not discussed here; see Stede, 1996a).

Morphosyntactic features. Standard features needed by the surface generator to produce syntactically correct utterances.

Similar words will share the same values for most dimensions and differ only in a few respects. For example, stylistic variants like *pass away* and *kick the bucket* differ only in terms of their connotations. More or less specific words have a different range of applicability, which is reflected in the denotation (in accordance with the subsumption relationships encoded in the DM). Other verbs differ in terms of their alternation behavior, as in the case of *give* (dative possible) versus *donate* (no dative) or in the *fill/füllen* example mentioned above. This example also coincides with a difference in *valency*, which is encoded in the PSemSpec and will be discussed in the next section.

The essential components for the SitSpec–SemSpec mapping are the denotation, covering list, and PSemSpec. Importantly, the denotation of a lexeme need not be a single concept; instead, it can be a complete configuration of concepts and roles (in the KL-ONE sense). This is necessary since we break up the internal event structure in the representation of verb meaning. The consequences are higher computational cost in finding lexical options, but also a higher flexibility in finding different verbalizations of the same event. As an example, consider the denotation of the causative reading of *fill* below, where concept names and atomic values appear in lower-case letters, and role names and variables in upper-case letters.

```
DEN: (event (PRE-STATE (fill-state (VALUE (not 'full))
                                   (CONTAINER A)))
          (ACTIVITY (CAUSER B))
          (POST-STATE (fill-state (VALUE < D 'full >)
                                   (CONTAINER A)
                                   (CONTENT C))))
```

The variables are bound to instances or atomic values of the SitSpec when the two are matched against each other. The filler of the `VALUE` role in the `POST-STATE` appears in angle brackets because it is a default value, which is filled in if no value for that role is present in the SitSpec. The accompanying partial SemSpec of causative *fill* contains the same variables:

```
PSS: (x / directed-action :lex fill
     :actor B :actee A :inclusive C <:destination D>)
```

In the first step of sentence generation, when the denotation is matched against a SitSpec, the variables get bound to SitSpec instances; in the later step of unifying lexical PSemSpecs into a sentence SemSpec, the variables in the PSemSpec are recursively replaced with PSemSpecs of the lexemes that cover the SitSpec instance that has been matched against the corresponding variable in the denotation. This procedure will be explained in Section 4. From the completed SemSpec, Penman then produces a sentence; for example *Jill filled the tank with oil.* (If the `VALUE` is different from `'full`, it also gets verbalized, such as in *Jill filled the tank to the second mark.*)

If any *selectional restrictions* are to be encoded as part of verb meaning, they can be added to the denotation. For instance, to restrict the CAUSER of *fill* to the type PERSON, the corresponding line of the denotation is changed to `(ACTIVITY`

`(CAUSER (PERSON B)))`. In this way, selectional restrictions of a verb can add more specific constraints to the type requirements already encoded in the domain model. The concept DIE, for example, can in the DM be restricted to objects of type LIVING-THING, and specific verbs can further restrict their applicability to certain subtypes of LIVING-THING, such as PERSON for *pass away*, or ANIMATE-BEING for *perish*.

By means of sharing the variables between denotation and PSemSpec, the lexicon entries work as a "bridge" between the SitSpec to be verbalized and the intermediate representation SemSpec. The connotations and salience assignments serve to establish an ordering of the lexemes that are considered when constructing a SemSpec; we do not discuss the specifics here.

When verbalizing SITUATIONS, the root node of the SitSpec is usually covered by the denotation of a verb. Hence, verb semantics plays a central role in the system.

3. Verb Valency and the Upper Model

3.1. Valency

To describe the role of the verb in the sentence (and, specifically, in sentence production), we make use of the notion of *valency*. The term is usually attributed to Tesnière (1959), who was concerned with describing a verb's property of requiring – or tolerating – certain arguments beside it in the sentence. Tesnière made the fundamental distinction between *actants* and *circumstantials*: the former are central participants in the process denoted by the verb, while the latter express the associated temporal, locational, and other circumstances. Three actants were distinguished and linked to functional elements: subject, direct object, and indirect object. But at the same time, Tesnière also linked the actant–circumstantial distinction to syntactic form and proposed an association between actants and nominal phrases, and between circumstantials and prepositional phrases.

While the actant–circumstantial distinction is in principle widely accepted, views differ on where to draw the line and how to motivate it. The notion of valency was further developed predominantly in German linguistics, with a culmination point being the valency dictionary of German verbs by Helbig and Schenkel (1973). They made an additional distinction between "obligatory" and "optional" actants (which they called "complements"); Somers (1987, ch. 1) proceeded to propose six different levels of valency binding. He also reviews a range of linguistic tests that were suggested by various researchers to determine the valency binding of sentence constituents.

Besides the problem of exactly distinguishing different kinds of valency binding, there are different opinions on the *type* of entities that are subject to a verb's valency requirements. Helbig and Schenkel (1973) characterize complements (actants) in terms of syntactic class, Leech (1981) raises the idea of "semantic valency" to operate on a level different from surface syntax, and Allerton (1982) emphasizes the importance of functional description for defining valency (i.e. in terms of subject, object, etc.). In short, there are quite distinct possibilities for dealing with the phenomenon.

In our approach, which is driven by the (practical) needs of MLG, we aim at encapsulating syntactic matters in the front-end generators and deal with valency in the SitSpec–SemSpec mapping: when characterizing the linking between SitSpec elements and SemSpec participants/circumstantials, we describe valency in terms of UM concepts (essentially a variant of “deep cases”).

We wish to distinguish cases like the following:

– *Tom disconnected the wire* {*from the plug*}. *Disconnect* requires a SOURCE, but it can be omitted in a suitable specific context.
– *Sally ate*. While *eat* usually requires a direct object, it can also be used intransitively due to the strong semantic expectation it creates on the nature of the object, *independent* of the context.
– *Tom put the book on the table. Put* requires a DESTINATION, and it cannot be omitted, no matter how specific the context.
– *The water drained* {*from the tank*} {*into the sink*}. *Drain* requires some PATH expression, either a SOURCE, or a DESTINATION, or both. But (in this reading) it cannot occur with no PATH at all.
– *The water drained from the tank* {*in the garage*}. Locative circumstances like *in the garage* are not restricted to particular verbs and can occur in addition to PATHS required by the verb.

Adopting the three categories proposed by Helbig and Schenkel (1973), we distinguish between obligatory and optional participants on the one hand, and circumstances on the other. The criterion of optionality, as indicated above, singles out the obligatory complements. But how, exactly, can we motivate the distinction between optional participants and circumstances in our framework? By relating the PSemSpec to the SitSpec, via the denotation. In the *disconnect* case, for instance, the two items CONNECTOR and CONNECTEE are both integral elements of the situation. The situation would not be well-formed with either of them absent, and the DM encodes this restriction. Therefore, both elements also occur in the denotation of *disconnect*, and a co-indexed variable provides the link to the PSemSpec. Only when building the sentence SemSpec is it relevant to know that the CONNECTEE can be omitted. The CONNECTEE in the denotation therefore must have its counterpart in the PSemSpec – that is the SOURCE, but there it is marked as optional (see Figure 5 below).

With circumstances, the situation is different: a SitSpec is complete and well-formed without the information on, for instance, the location of an event. Hence, a verb’s denotation cannot contain that information, and it follows that it is not present in the PSemSpec, either.

3.2. VERBS AND THE UM

Now, since our instrument for ensuring the well-formedness of PSemSpecs and SemSpecs is the UM, we need to inspect the role of valency information in it. On the one hand, Bateman et al. (1990) are well aware of the problems with ascribing simple valency patterns to verbs, but for the practical implementation of Penman and the UM, some strict – and simplifying – category distinctions had to be made. Thus, all participants of process types, as listed above, are coded in LOOM as obligatory roles. Furthermore, for specific process types, the roles can be value-restricted. Circumstances, on the other hand, are in the UM coded as LOOM relations, and there are no restrictions as to what circumstances can occur with what processes. In general, any spatio-temporal information is seen as a circumstance. Concerning the linguistic realizations, Penman and the UM in their present form essentially go back to the Tesnèrian suggestion that participants are realized as nominal groups (with some obvious exceptions, as in *say that x*), and circumstances as prepositional phrases or as adverbs.

Indeed, many spatio-temporal aspects of situations can be clearly classified as circumstances, and they are syntactically realized as just predicted: something happened *yesterday/on Monday*, and it occured *in the city*. But neither the syntactic division corresponding to participants and circumstances (direct or indirect object versus adverbs or prepositional phrases) nor the UM's semantic postulate that spatio-temporal aspects are circumstances hold *in general*. Regarding spatial relationships, we find verbs that specifically require *path*-expressions, which cannot be treated on a par with circumstances: recall *put*, which requires a direct object and a destination. Causative *pour* requires a direct object as well as a path with either a source, or a destination, or both: *pour the water from the can into the bucket*. Some verbs, as is well known, can occur with either a path (*Tom walked into the garden*) or with a place (*Tom walked in the garden*), and only *in the garden* can here be treated as a circumstance. And *disconnect* requires a direct object (the entity that is disconnected) and a source-expression (the entity that something is disconnected from). The source can be omitted if it is obvious from the context: *Disconnect the wire!* But it does not have the status of a spatial circumstance like *in the garage*.

The UM in its present form cannot make distinctions of this kind. It is not possible to specify a PATH expression, which will be realized as a prepositional phrase, as an obligatory participant, and it is not possible to represent the difference in valency for *walk* in *walk in the garden*/*walk into the garden*. About *disconnect* (in the causative reading), which is a MATERIAL-PROCESS, the UM can only state that the roles ACTOR and ACTEE must be filled, but not the fact that there is another entity involved – in the DM we called it the CONNECTEE – which can be verbalized as a SOURCE. Moreover, the UM does not know that the CONNECTEE is optional in the verbalization; it does not distinguish between obligatory and optional participants.

As a step forward to a more fine-grained distinction between participants and circumstances, we differentiate between requirements of process types (as coded in the UM) and requirements of individual verbs, which are to be coded in the lexical entries. In a nutshell, valency (as a lexical property) needs to supplement the participant/circumstance requirements that can be stated for types of processes. To encode the valency information, we use the partial SemSpec of a lexicon entry. The participant roles stated there are either obligatory or optional, in which case they are marked with angle brackets:

```
to disconnect
PSS: (x / directed-action
        :actor A :actee B < :source C >)
```

With obligatory participants, the verb is only applicable if the elements denoted by these participants are present in the SitSpec. Optional participants need not necessarily be included in the verbalization: if they are present in the SitSpec, they may be omitted if there is some good reason (e.g. a stylistic preference); if they are not present in the SitSpec, the verb can be used anyway.

Circumstances do not appear in the case frame but are added by general rules. To illustrate our treatment of valency and argument linking, Figure 5 shows excerpts from lexical entries of nine different verbs. The information is arranged as follows: on the right-hand side is the case frame of the verb, written as the SemSpec participant keywords (each starting with a colon). Optional participants are enclosed in angle brackets. On the left-hand side are excerpts from the denotation: the names of the roles whose fillers are co-indexed with the respective position in the case frame. Thus, the arrows give the argument linking for the base form of the verb, which can be quite simple, as in *open* or *move*. From the perspective of the DM, the roles on the left-hand side of the arrows are required to be filled – as is encoded in the LOOM definitions of the underlying concept. Only items appearing with an asterisk in front of them are optional in the SitSpec: for example, a SitSpec underlying an OPEN event is well-formed with or without a CAUSER being present. These items can be added to the SemSpec by alternation/extension rules (which we have not explained here); the names of the applicable rules for a verb appear below the line.

4. Mapping SitSpecs to SemSpecs

Having discussed the various declarative representations involved, we now turn to the actual mapping procedure that produces a language-specific SemSpec from a SitSpec. We will first explain the step of finding lexical options, and then the one of constructing a SemSpec; as indicated earlier, for reasons of brevity we omit the phase of finding verb alternations/extensions and the evaluation of preferential features (which are both not critical for presenting the basic mechanism). To illustrate the procedure, we use the example of Jill filling a tank with water, the SitSpec of which was given in Figure 3, and some SemSpecs in Figure 4. Specifically, we will explain the production of the (simple) sentence *Jill filled the tank with water*.

DISCONNECT CAUSER → :actor CONNECTOR → :actee CONNECTEE → <:source>	**OPEN** OBJECT → :actor *CAUSER ——— resultative-causative
POUR PATH-SOURCE → :actor OBJECT → <:actee> *PATH-DESTINATION *CAUSER ——— substance-source durative-causative	**SPRAY** CAUSER → :actor OBJECT → :actee PATH-DESTINATION → :destination ——— spray-load
DRAIN OBJECT → :actor PATH-SOURCE → <:source> *PATH-DESTINATION *CAUSER ——— durative-causative locative/clear-intransitive resultative-causative	**FILL** CONTENT → :actor CONTAINER → :actee VALUE → <:destination (default)> *CAUSER ——— stative-resultative resultative-causative
MOVE/WALK OBJECT → :actor *PATH *CAUSER ——— durative-causative	**LEAK** PATH-SOURCE → :actor OBJECT → <:actee> *PATH-DESTINATION ——— substance-source

Figure 5. Excerpts of sample lexical entries for verbs.

4.1. Find Lexical Options

If language generation is based on a rich dictionary, offering an array of synonymous or nearly synonymous lexical items for expressing a certain concept, lexical decisions will interact not only with one another, but also with many other decisions to be made by the system. In order to be able to account for as many of these inter-dependencies as possible, we take the first step in the generation process to be the determination of *verbalization options*: the set of all words or phrases that can express some part of the proposition to be expressed. In effect, the set of these options constitutes the search space for planning the structure of the sentence.

Technically, determining the set of lexical options amounts to finding those lexemes whose denotation subsumes some part of the SitSpec. That is, for every node I in the SitSpec, we want to find all lexical items whose denotation is in a *subsume* relationship to the subgraph rooted in I. We thus make use of our taxonomic background knowledge base with the domain model, in which word meaning is grounded.

The first condition for *subsume* is that the type of the denotation's root node is more general than, or the same as, the type of I's root. Next, the denotation root must not have a role associated to it that I does not have; otherwise, the lexical item would imply more than is warranted by the proposition. But conversely, I may very well have roles that are not defined for a lexical item, yet the item may be appropriate; in this case, the item is more general, i.e. it conveys *less* than warranted by the proposition, which, for some reason or another, might be desired. Finally, since we have to match paths of (in principle) arbitrary length, *subsume* must also hold recursively between role fillers. For all of I's roles, if they are also defined for the candidate denotation, then *subsume* most hold between the role filler of the denotation and the role filler of I.

More formally, the function *subsume* can be described as follows. Let I denote the SitSpec node under consideration, and $t(I)$ a function that returns the type of I. $C_1 \succeq C_2$ denotes concept subsumption in the domain model, i.e. C_1 is more general than C_2;[4] $R(i_1, i_2)$ means that relation R holds between two nodes (in other terminology, i_2 fills role R of i_1). Then we are looking for the set of lexemes with denotations i such that:

$$\begin{aligned} subsume(i, I) \;\Leftrightarrow\; & \; t(i) \succeq t(I) \\ & \wedge \forall R \,\forall x\, [R(i,x) \rightarrow \exists y\, [R(I,y) \wedge subsume(x,y)]] \end{aligned}$$

The actual matching procedure in MOOSE is slightly more elaborate, because the syntax of denotations and SitSpecs is not exactly the same. The procedure has to account for variables in the denotation and bind them to the SitSpec nodes they match; if there is a type restriction associated with a variable, it has to be ensured that the SitSpec node is in fact subsumed by that type. And finally, if the denotation contains a default value, then the role need not be present in the SitSpec for the matching to succeed, but if it is present, subsumption must hold, as for any other role filler.

The matching procedure is executed for every node of the SitSpec, in order to determine the lexemes that can potentially cover that node. But instead of blindly searching the entire lexicon at every SitSpec node, only those entries are tested whose denotation root node has either the same type as the SitSpec node or a more general one. In the implementation, this indexing is performed by exploiting the functionality of LOOM. Also, all $\succeq$ checks mentioned above are performed by LOOM.[5]

In effect, we are defining the relationship between words and concepts as one variant of "structural subsumption" in the terms of Woods (1991, Section 1.5.5). While previous generation systems have typically used simple associations between words and single (atomic) concepts, our approach of comparing complex word denotations to conceptual configurations allows for producing variation in lexical specificity, such as those discussed in Section 1. Furthermore, an elaborate word–concept mapping is a prerequisite for variants of incorporating units of meaning in lexical items in different ways.

If the denotation of a lexical entry *LE* thus subsumes a SitSpec node, it becomes *instantiated* and is added to the pool of verbalization options *VO*. Instantiation means that the names of nodes in the denotation and covering-list of *LE* are replaced by the names of the SitSpec nodes they match and that variables are bound to the SitSpec nodes they match. This binding is propagated from the denotation to the same variable in the PSemSpec of *LE*. Simultaneously, a backward pointer is established from the SitSpec node to the *vo*: each SitSpec node has a "covered-by" list associated with it, and the name of the *vo* just formed is added to this list. Thus, after the matching phase, the exact covering relationships between *VO* and SitSpec are recorded on both sides. The "covered-by" lists associated with the SitSpec nodes will later be used to drive the SemSpec construction.

Example. In our example Jill filled the tank, the matching phase finds, *inter alia*, the lexical entries shown in Figure 6 (for brevity, we list only the denotation, the PSemSpecs, the pointers to alternations/extensions, and the covering-lists). As can be seen when comparing to the SitSpec, the denotations match the respective SitSpec nodes. The entries now get instantiated: instance names replace concept names in the `DEN` and `COV` components.

4.2. Determine the Complete and Preferred SemSpec

Omitting the step of constructing alternations, the next task is to find a subset VO' of the verbalization options such that:

1. The partial SemSpecs associated with the elements of VO' can be unified into a single, well-formed SemSpec.
2. The elements of VO' collectively cover the entire SitSpec, i.e. every element in SitSpec will be expressed in the sentence.
3. No element of the SitSpec is expressed more than once.
4. The resulting SemSpec is *preferred* in a weak sense; see below.

Well-formedness (1) of the SemSpec is guaranteed by constraints specified in the UM concepts (the names of which appear in the partial SemSpecs of the lexical options). Specifically, the PROCESS concepts in the UM define which participants

```
to fill (stative)
DEN: (fill-state
        (CONTENT A)
        (CONTAINER B)
        (VALUE < C 'full >))
PSS: (x / directed-action
       :lex fill_el
       :actor A
       :actee B
     < :destination C >)
AER: ((stative-resultative
     resultative-causative))
COV: (fill-state < 'full >)
```

```
jill
DEN: (jill)
PSS: (x / person
       :name jill)
COV: (jill)
```

```
water
DEN: (water)
PSS: (x / substance
       :lex water_el)
COV: (water)
```

```
tank
DEN: (tank)
PSS: (x / object
       :lex tank_el)
COV: (tank)
```

Figure 6. Lexical entries matching the SitSpec in *fill* example.

are obligatory, and what UM type the participants must belong to. A well-formed SemSpec must be *saturated*, that is, all variables (if any) must have been replaced with PSemSpecs, which are themselves saturated. Requirement (2) makes sure that no element of the SitSpec is excluded from the verbalization. Requirement (3) ensures that the resulting sentence is *minimal* in the sense that the generator does not produce a sentence like *We drove by car*, which covers the INSTRUMENT with both the verb and a prepositional phrase. The factors for preference evaluation (4) are not our topic here, as mentioned earlier. But, very briefly, what the system does is first to compute a local order of verbalization options for each SitSpec node, and then to consider at every node the options in that order. This is not guaranteed to find the *overall* best solution, but works quite well in the majority of cases.

The procedure "BuildSemSpec" is given in Figure 7. It takes as arguments a SitSpec node – the one that a SemSpec is to be built for – and the ordered list of verbalization options *VO* that cover this node (recall that in the matching phase the *vo*s were placed on the "covered-by" list of the respective SitSpec nodes). When verbalizing a SitSpec, we make use of the obvious fact that the SemSpec we are looking for must cover the root node of the SitSpec. We thus apply the BuildSemSpec procedure to the root node, consider the PSemSpecs associated with it and try to saturate one of them, in their order of preference. As soon as one can be saturated, which also covers the complete SitSpec (except possibly for optional nodes), we have the result.

BuildSemSpec returns a vector of two components, `result.semspec` and `result.covering`. The procedure works as follows. The current *vo* is set to the most preferred one (line 2). If the PSemSpec component of the *vo* is already saturated, we are finished and return the result vector for the current *vo* (line 4). Otherwise, every external variable in the PSemSpec needs to be replaced by a saturated SemSpec. In line 6, the procedure CorrespondingNode is called; it determines the SitSpec node that corresponds to the external variable shared by *vo*`.psemspec` and *vo*`.denotation`. This is the SitSpec node that needs to be processed in order to replace the current external variable; line 7 has the recursive call of BuildSemSpec. If the result is a well-formed SemSpec, the external variable is replaced with that SemSpec (line 13), and the covering list of the *vo* is extended

```
Procedure BuildSemSpec(sitspecnode, [vo_1, ..., vo_n])
1 i := 1
2 L1: vo := vo_i
3    IF Saturated(vo.psemspec)
4      THEN RETURN [vo.psemspec, vo.covering]
5      ELSE FOR EVERY ext_var IN vo.psemspec
6             newnode := CorrespNode(ext_var, vo.denotation, sitspec)
7             result := BuildSemSpec(newnode, newnode.vo)
8             IF result = 'fail' OR UM-Incompatible(result.semspec)
9               THEN IF i = n
10                     THEN RETURN 'fail'
11                     ELSE i := i + 1
12                          GOTO L1
13              ELSE ext_var := result.semspec
14                   vo.covering := vo.covering ∪ result.covering
15          RETURN [vo.psemspec,  vo.covering]
```

Figure 7. The procedure for building SemSpecs (simplified).

with the covering list determined by the recursive call (line 14). If, on the other hand, the recursive call did not succeed in finding a saturated SemSpec, or if the result does not respect the constraints imposed by the UM (line 8), we need to backtrack (lines 9–12). If we have further *vo*s available, then the next one is tried; otherwise the procedure has to return `'fail'`.

Once BuildSemSpec was applied to the root node of the SitSpec and has produced a SemSpec and a covering list, this covering list is compared to the list of SitSpec nodes. If one or more nodes are uncovered, the procedure backtracks to the next possible solution.

Example. Returning to the example Jill filled the tank, the “covered-by” list of the SitSpec node `event-1` contains, amongst others, the *vo* `fill3` (shown in Figure 8), which was produced by an alternation rule. Assuming here that this option is first in the list, the BuildSemSpec procedure inspects the PSemSpec associated with the *vo* and has to handle the external variables `Y`, `A`, and `B` in turn. These variables are looked up in the denotation, and the corresponding nodes in the SitSpec are determined, which are `jill-1`, `water-1`, and `tank-1`, respectively. For each node, the procedure calls itself and immediately returns the PSemSpecs associated with the nodes (see Figure 6), because they are already saturated. Thus, they replace the variables, no further recursion is necessary, and the final SemSpec is as shown below, together with the covering list that results from merging the PSemSpecs.

```
(x1 / directed-action :lex fill_el
  :inclusive (x2 / substance :lex water_el)
  :actee (x3 / object :lex tank_el)
  :actor (x4 / person :name jill))

  COV: (fill-state-2 'full pour-1 event-1 jill-1 water-1 tank-1)
```

```
fill2 (stative-resultative of fill)
DEN: (event-1 (ACTIVITY pour-1)
              (POST-STATE
              (fill-state-2 (CONTENT A)
                            (CONTAINER B)
                            (VALUE 'full))))
PSS: (x / nondirected-action :lex fill_el
        :inclusive A
        :actor B)
COV: (fill-state-2 'full pour-1 event-1)
```

```
fill3 (resultative-causative of fill2)
DEN: (event-1 (ACTIVITY (CAUSER Y)
              (POST-STATE
              (fill-state-2 (CONTENT A)
                            (CONTAINER B)
                            (VALUE 'full))))
PSS: (x / directed-action :lex fill_el
        :inclusive A
        :actee B
        :actor Y)
COV: (fill-state-2 'full pour-1 event-1)
```

Figure 8. Instantiated verbalization options in *fill*–example.

5. Lexical Paraphrases

Having explained the mechanisms, we now describe the range of paraphrases that MOOSE produces by constructing different SemSpecs from the same SitSpec. We give examples for both monolingual and multilingual variation.

5.1. INCORPORATION

It is well known that meaning can be distributed to words across the sentence in various ways, by means of different incorporations. Elementary units of meaning can either be expressed separately, or “swallowed” by another word, as in *affect adversely* / *impair*, or *move upward* / *rise*. On the multilingual side, the seminal work by Talmy (1985) demonstrated that different languages (or language families) can exhibit different tendencies for incorporating information, what he called “lexicalization patterns”. English motion verbs, for example, tend to express the MANNER, whereas Romance languages prefer to incorporate the PATH into the verb: *The bottle floated into the cave* / *La botella entró a la cueva flotando* (‘The bottle entered the cave floating’), or *He swam across the river* / *Il a traversé la rivière à la nage* (‘He crossed the river swimming’).

In order to achieve variation of this kind in NLG, it is obviously necessary that the individual elements of meaning be explicitly represented, so that they can be mapped to lexemes in different ways; this task is then the sub-lexical version of the “chunking problem” in NLG (the problem of dividing input structures into individual units to be communicated). In MOOSE, the covering mechanism together with the subsumption check in denotation matching achieve this functionality. In the example *go by plane / fly*, the general verb *go* covers only the MOVE concept, and the role INSTRUMENT – PLANE is left to be expressed by a prepositional phrase; whereas the specific verb *fly* covers the whole configuration. Both are alternatives for verbalization, and a choice can be made on the grounds of desired brevity, salience assignment, etc.

5.2. CONNOTATION

Without going into detail here, we treat connotations as secondary aspects of word meaning that can distinguish words with the same denotation.[6] Differences in connotation also point to the somewhat related problem of *collocations* – affinities between lexemes that tend to occur together – which we have not dealt with at all.[7]

MOOSE follows the idea that connotations can be factored into a number of distinct features. Here is an example for words indicating that their referent belongs to a different "class": a person can have a *job* as a janitor, and another one an *appointment* as a professor; exchanging the combinations would lead to a rather ironic formulation. Very often, *idiomatic* variation is also a matter of changing connotations, because many idioms tend to convey a colloquial or maybe vulgar tone, like the notorious *kick the bucket*. Near synonyms of this kind have the same denotation and coverage in MOOSE and thus are always considered as verbalization alternatives; the choice is made by a distance function comparing the associated stylistic features with a target configuration.

As an example combining language-specific incorporation with the possibility of a stylistic choice, consider the following case from the multilingual automobile manual quoted in Section 1. At the end of the instructions for changing engine oil, we find the phrase *Replace the cap*, and the German version is *Den Deckel wieder anbringen* (lit. 'the cap again install'). In English, *replace* is ambiguous between a sense similar to *substitute* or *exchange* and one where the prefix *re-* indicates that a previous state is restored. The latter sense is the one needed here. In German, it is not possible to incorporate the restoration facet into the verb; instead it has to be expressed with an adverb like *wieder* ('again').

Given that the feature of restoration is coded in the SitSpec as a role attached to the activity INSTALL (which is not difficult, because a planning module would have the information on which actions reverse the effects of others and thus could add the role to the concept), then the incorporation/covering mechanism operates here in the same way. Only, for German there would be no incorporation option. There is a choice between two near synonyms, though: *wieder* is a regular, "core" adverb, and *abermals* is a more formal expression with the same denotation. For instance, if a speaker does not like to do something (e.g. install the oil filler cap), then he or she can indicate the displeasure of having to do it once more by using *abermals*. Contrary to *wieder*, this adverb tends to precede the direct object, a fact that the surface generator would be in charge of knowing. To place additional emphasis on it, it can be thematized. In short, the system can produce any of these variants:

Tom replaced the cap.
Tom installed the cap again.
Tom brachte den Deckel wieder an.
Tom brachte abermals den Deckel an.
Abermals brachte Tom den Deckel an.

MOOSE can make its choice on the basis of salience or, in German, of the connotations. (There is a similar distinction in English between *again* and *once more*, which we ignore here.)

5.3. SPECIFICITY

An important dimension of choice is the specificity of the word one uses to refer to an object (*poodle, dog, animal*) or a situation (*darn the socks, mend the socks*). The more general word has less information, in the sense that it can denote a wider class of entities; yet there can be good reasons for using it, for example when certain connotations are to be expressed. If one does not like the neighbor's dog, the derogatory attitude can be conveyed by referring to it as *that animal*, provided that *animal* is considered as a lexical option, which MOOSE does by finding candidates via subsumption.[8]

Lexical grain-size. Sometimes, languages exhibit different grain sizes in their vocabulary; that is, one language makes a distinction where another does not. French *savoir* and *connaître* correspond to German *wissen* and *kennen*, but in English both are covered by *know*. This is the general phenomenon of different lexical taxonomies between languages. A notorious example is *put*, where German typically uses one of the verbs *setzen, stellen, legen* ('set', 'stand', 'lay'), which add information about the shape of the objects in question, and their relative positioning.

Provided that the crucial units of meaning correspond to concepts in the DM, which is a matter of granularity of representation, MOOSE handles lexical gaps with the subsumption checking in the matching phase: If there is a specific word in one language but not in the other, then a more general word is found and an appropriate modifier added, as in the cases explained above.

Language-specific conventions. The absence of some specific word from a language is one thing; a different matter is a tendency to use specific words less often. We mentioned in section 1 that English prefers to use abstract and less specific verbs where German has a concrete and quite specific verb. Recall the example of *remove* corresponding to numerous German verbs that specifically characterize the physical activity and the topological properties of the objects involved. These verbs are not absent in English, but it seems to be more common to describe the abstract effect of the action, as a matter of convention.

Again, since SitSpec – denotation matching goes not for identity but for subsumption – the range of more or less specific lexemes for a given concept is considered automatically. This is straightforward in the case of nouns denoting single concepts, but also works for verbs and events. The subsumption relationships encoded in the domain model determine what is recognized as a paraphrase of this kind and what is not; let us consider an example in detail.

We mentioned earlier the situation of Jill uncorking a wine bottle. Concentrating on English examples, we demonstrate how the following verbalizations are produced:

Jill opened the bottle.
Jill removed the cork from the bottle.
Jill uncorked the bottle.

The SitSpec is shown in Figure 9. To understand the connections between the verbalizations, an excerpt from the domain model in Figure 10 shows the subsumption relationships between LOCATION-STATE, CONNECTION-STATE, and TANK-OPEN-STATE. They are all specializations of TERNARY-STATE, a concept subsuming all states composed of three different entities (as opposed to BINARY-STATES, which relate a single attribute to a value, hence involve two entities). For our purposes, we assume that the concept BOTTLE is a specialization of TANK.

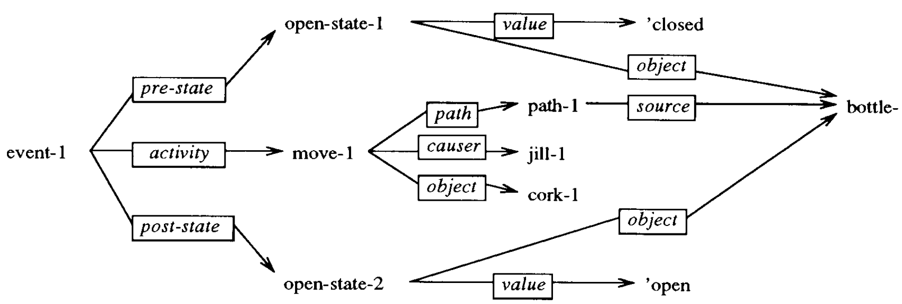

Figure 9. SitSpec for Jill uncorking the bottle.

Consider first *open*, and assume that the alternation rules have already computed its causative reading:

```
NAM: open
DEN: (event (PRE-STATE (V1 open-state (CONTAINER A)
                                      (VALUE 'closed)))
             (ACTIVITY (V2 (CAUSER B)))
             (POST-STATE (V3 open-state (CONTAINER A)
                                      (VALUE 'open))))
COV: (event V1 V2 V3 'closed 'open)
PSS: (x / directed-action :lex open_el :actor B :actee A)
```

Both *remove* and *uncork* are inherently causative, i.e. their base forms already contain a CAUSER. *Remove* (see below) denotes that somebody moves a LOCATUM, which had occupied a LOCATION, so that afterwards the LOCATUM is no longer in the LOCATION. Importantly, the node `(not B)` in the POST-STATE is covered by this verb, because that is exactly what it expresses: move something *away from* a location.

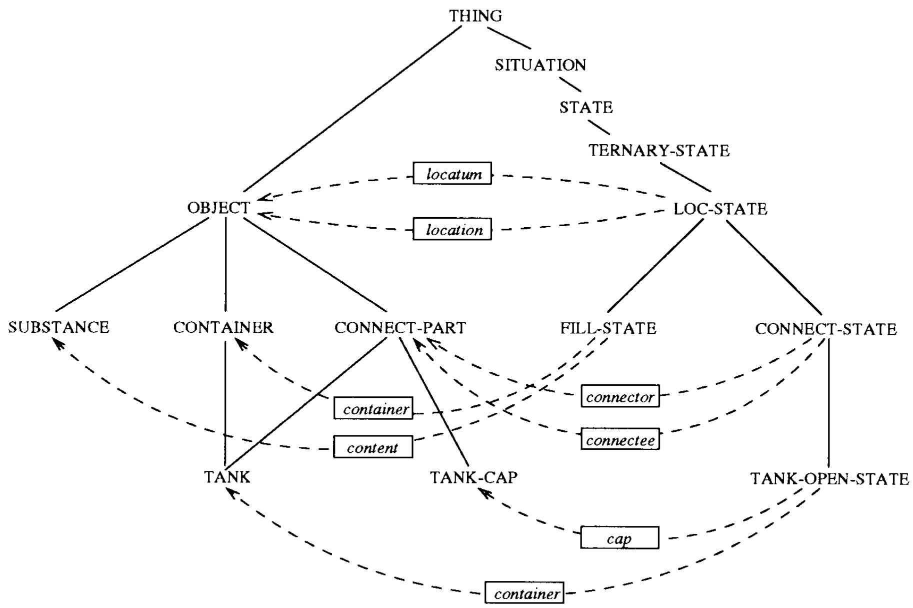

Figure 10. State subsumption in domain model (excerpt).

Uncork is a specialization of *remove*, as far as the denotation is concerned, but the case frame is quite different. Notice first the selectional restriction: *uncork* is a way of removing that applies only to corks in the openings of bottles, hence the denotation contains appropriate selectional restrictions. And the verb also covers the cork, because the fact that *a cork* is removed is an inherent part of the meaning of *uncork*. As a consequence, the LOCATUM cannot occur in the case frame; instead the `:actee` must be the bottle, and there is no optional `:source`.

```
NAM: remove
DEN: (event (PRE-STATE (V1 location-state (LOCATUM A)
                                          (LOCATION B)))
            (ACTIVITY (V2 (CAUSER C)))
            (POST-STATE (V3 location-state (LOCATUM A)
                                          (LOCATION (not B)))))
COV: (event V1 V2 V3 (not B))
PSS: (x / directed-action :lex remove_el
         :actor C :actee A  < :source B >)

NAM: uncork
DEN: (event (PRE-STATE (V1 location-state (LOCATUM (cork A))
                                          (LOCATION (bottle B))))
            (ACTIVITY (V2 (CAUSER C)))
            (POST-STATE (V3 location-state (LOCATUM (cork A))
                                          (LOCATION (not B)))))
```

```
COV: (V1 V2 V3 A (not B))
PSS: (x / directed-action :lex remove_el :actor C :actee B)
```

When comparing these denotations to the SitSpec, with the subsumption relationships defined in Figure 10 in mind, it becomes clear that indeed the denotations of all the verbs match the SitSpec. After their instantiation and SemSpec unification, they produce the three sentences we have listed above. After all, TANK-OPEN-STATE is subsumed by LOCATION-STATE, and the roles are also in the required subsumption relationships.

Imagine now that an additional attribute were attached to the CORK in the SitSpec, for example the information that it is moldy. Assuming that the only way of verbalizing this attribute is an adjective, then there is only one sentence to describe the situation: *Jill removed the moldy cork from the bottle.* Sentences that use *open* and *uncork* will still be correct and well-formed, but the system notices that they do not cover the node MOLDY; hence they are incomplete and will not be considered further.

While opening up the door to inheriting more general lexemes and considering them for generation is very important, it also creates a new problem: that of *stopping* inheritance. In the place of *Tom sat on the chair*, one normally should not even consider an utterance like *Tom sat on the object*. MOOSE eschews these problems using the preference mechanism: it assigns higher scores to more specific lexemes, and only when other preferential factors favor a more general word, it is used. But the theoretical question of when to block lexical inheritance and when to allow it, is an open issue.

5.4. EMPHASIZE AN ASPECT

When describing an event, a verb can denote a certain aspect of that event and leave other aspects to be inferred by the hearer. An example from Wunderlich (1991): a roll-up shutter, as sometimes used on shop windows and doors, can be *opened* in English, but hardly ?*drawn up* or ?*pulled up*. In German, however, the word corresponding to *open* (*öffnen*) is quite uncommon in this context; it is more natural to use *hochziehen* ('to draw up') or *aufziehen*. Morphologically, the latter can be either an amalgamation of drawing and opening, or a shorthand for *heraufziehen*, which is a synonym of *hochziehen*. Similarly, in French one uses *tirer en haut* ('to draw upwards'). Thus, English prefers to verbalize the result of the action, while German and French characterize the action itself. This corresponds to the observation regarding the ubiquity of *remove* mentioned above.

As a detailed example for the difference between focusing on the activity and on the state transition, we go back to another phrase from the automobile manual:

Disconnect the wire from the plug
Ziehen Sie das Zündkabel von der Zündkerze ab.
('Pull the wire off the spark plug')

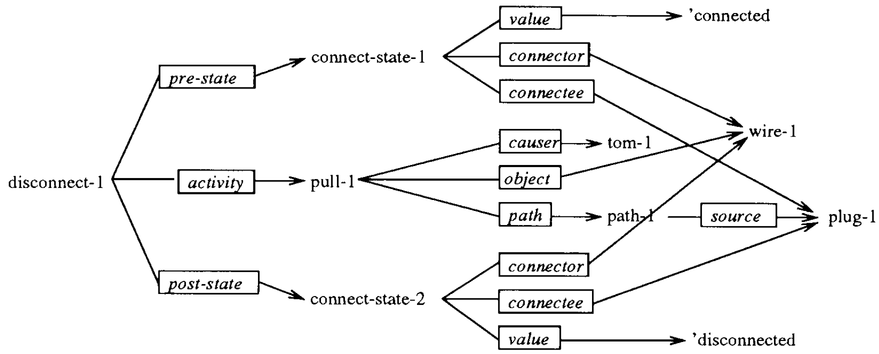

Figure 11. SitSpec for Tom disconnecting the wire.

Again, the English version characterizes the technical effect of the event, whereas the German one describes the physical activity leading to it. Figure 11 gives the SitSpec underlying both verbalizations.

The lexical entry for *disconnect* is a complex one, because the verb in its most basic form already expresses a state change as well as the fact that some CAUSER is responsible.

```
NAM: disconnect
DEN: (event (PRE-STATE (V1 connection-state
                            (CONNECTOR A)
                            (CONNECTEE B)
                            (VALUE V2 (not 'disconnected))))
             (ACTIVITY (V3 (CAUSER C)))
             (POST-STATE (V4 connection-state
                            (VALUE 'disconnected))))
COV: (event V1 V2 V3 V4 'disconnected)
PSS: (x / directed-action :lex disconnect_el
         :actor C :actee A < :source B >)
```

It covers the PRE-STATE and the POST-STATE, as well as the ACTIVITY and the VALUES of the states. The SemSpecs of the remaining elements replace the variables in the PSS, and the sentence results, either with or without mentioning the SOURCE, because it is optional in the case frame. Notice that all the SitSpec nodes are indeed covered, because the elements of the ACTIVITY are also elements of the STATES, except for the CAUSER, which is contained in the case frame of the verb.

In German, *ziehen* is largely translation-equivalent to *pull*; it denotes only the ACTIVITY from the SitSpec. It does, however, undergo extensions that morphologically add the prefix *ab-* to the verb (similar to the particle *off* in English) and then it denotes in addition the fact that the object pulled is afterwards disconnected from its original location, hence *abziehen* also covers the POST-STATE. Our system does not handle morphological derivation, though, so we treat *abziehen* as a separate entry.

```
NAM: abziehen
DEN: (event (PRE-STATE (V1 connection-state
                               (CONNECTOR A)
                               (CONNECTEE B)
                               (VALUE (not 'disconnected))))
             (ACTIVITY (pull (CAUSER C)
                               (OBJECT A)
                               (PATH (SOURCE B))))
             (POST-STATE (V2 connection-state
                               (CONNECTOR A)
                               (CONNECTEE B)
                               (VALUE 'disconnected))))
COV: (event pull V1 V2 'disconnected)
PSS: (x / directed-action :lex abziehen_gl
        :actor C :actee A < :source B >)
```

The case frame is analogous to that of *disconnect*, and the variables are replaced in the same way, so that we arrive at the SemSpec for *Tom zog das Zündkabel von der Zündkerze ab.*

Emphasizing a different aspect can also lead to quite different sentence structures, and between different languages this is often not a matter of choice. In Section 1, we gave the example *Twist the cap until it stops* (two clauses linked by a connective) and the German variant *Drehen Sie den Deckel bis zum Anschlag* (lit. 'turn the cap up to the stop', one clause with a prepositional phrase). The common underlying SitSpec needs to express that a connection changes its state from "loosely connected" to "tightly connected" and that this transition is brought about by a turning activity that the reader executes on the cap, up to the point where it cannot continue. To this end, *twist* needs to be decomposed into movement along a path, specifically along a circular path, where something moves around an axis.

The central difference between the English and the German verbalization is that the former views the event as one process whose termination is defined by another process. The sentence structure results from the lexeme *until* becoming the element that covers the relationship between a durative activity (of which MOVE is a sub-concept) and a goal state. The German utterance, on the other hand, expresses only a single process, whose termination is (optionally) subcategorized for by the verb *drehen*. English *twist* cannot subcategorize for an equivalent of *Anschlag* ('stopping point'), which is the reason for the incongruity between the examples. Hence, *drehen*, denoting movement along a circular path, becomes the head of a single clause.

6. Summary and Comparison to Related Research

Lexical choice. Lexical choice has traditionally not been a strength of natural language generators. While a wide range of choice criteria have been investigated in isolation (see Stede, 1995 for an overview), no unifying system for making the overall best possible word choices has been found: we are still quite far away from what Busemann (1993) labeled “holistic lexical choice”.

MOOSE was designed to provide a framework in which a number of diverse factors influencing lexical choice can be accommodated. By separating the constraints (the denotation of a lexeme must subsume part of the SitSpec) from the preferences (connotations, salience) in the declarative representations and the lexicalization mechanism, a wide variety of lexical paraphrases can be systematically produced. As a prerequisite, MOOSE first computes the range of possible verbalization options and then tries to find the most suitable combination; the first step is a development of the approach taken by Miezitis (1988), who worked in a framework of marker passing and matching based on numerical scores, rather than on subsumption. Preferential word choice, as in our second step, has earlier been used in DIOGENES (Nirenburg and Nirenburg, 1988), but there the choice occurred *only* by computing preferences; no distinction between denotation and connotation, hence between constraints and preferences, was made.

When finding lexical options, the denotations and SitSpec parts to be compared are not single concepts but complex entities; this is a necessity when fine-grained lexical differences are to be accounted for. Horacek (1990) and Nogier and Zock (1992) made similar proposals of subgraph matching in the determination of lexical candidates. Our emphasis on the interplay between domain knowledge and lexical knowledge is akin to Jacobs’s (1987) work on the KING system, but there a single knowledge base holds all the necessary knowledge, linguistic as well as non-linguistic. In contrast, MOOSE emphasizes the separation of the different knowledge sources in order to enhance modularity and to enable multilingual generation.

Multilinguality. The smooth extension from monolingual to multilingual generation is in fact a central concern of the approach. Following the idea of keeping apart the language-specific knowledge where necessary but sharing the language-neutral knowledge where possible (as proposed for UMs by Bateman et al., 1991), MOOSE adds the emphasis on rich lexical representations and their central role in building semantic specifications from a “deeper”, non-linguistic representation. In this way, the system pursues the goal of viewing multilingual sentence generation as essentially the same task as monolingual paraphrasing.

Paraphrases. The availability of a range of paraphrases is a prerequisite for language generation to make intelligent choices that suit the particular audience or utterance situation. Our concern for paraphrasing was shared by Elhadad

(1993), whose ADVISOR II system generates short paragraphs advising students on whether to take particular courses. One dimension of lexical choice is treated in detail, namely that of argumentative intent. The central difference between Elhadad's and our approach is that the former opts for encoding all the paraphrasing power into the actual generation grammar, whereas we have emphasized the utility of a separate mapping step, prior to activating the front-end generation grammar. A modular architecture of this kind lends itself to multilinguality much more easily.

Another system that is strong on the paraphrasing side is GOSSiP (Iordanskaja et al., 1991). Rooted in the linguistic theory of the Meaning-Text Model (MTM), the emphasis is on deriving paraphrases in the semantics-to-syntax mapping, such as those based on support verbs (e.g. *to use Emacs / to make use of Emacs*). The MOOSE approach focuses rather on the concept-to-semantics mapping and thus addresses the problem from a different angle.

Two-level semantics. From the perspective of generation architecture, the central point made in MOOSE is the treatment of meaning on two distinct levels, and having the lexemes of a target language map between these levels. In linguistics, the advantages of two-level systems have been pointed out in particular for dealing with polysemy (Bierwisch, 1983; Pustejovsky, 1991b). Although we have not looked at these matters here, they were in fact an important factor in making the design decisions for MOOSE.

Our approach is similar to that of Dorr and Voss (1994), who propose a framework for interlingual MT, in which word meaning of source and target language is grounded in a KL-ONE knowledge base, and subsumption is exploited for dealing with lexical mismatches. Their system uses a single ontology and knowledge base, though, whereas we have – from the perspective of MLG – stressed the need to distinguish domain model and linguistic knowledge. Separating SitSpec from SemSpec, and assuming distinct ontologies underlying the two, has its advantages for language generation. Generation constraints can be stated on both levels, and for different purposes: applicability conditions for lexemes with respect to the domain model (in the denotations), and the contribution a lexeme makes to sentence meaning (in the PSemSpecs). Issues of compositionality can be confined to the SemSpec level. Word meaning can be grounded in background knowledge. Keeping these realms distinct also pays off when it comes to transferring the system to new domains: ideally, all components of lexical entries except for denotations will stay the same. In particular, PSemSpecs should not change, because a word exhibits the same behavior towards other words in the sentence, even if the meaning has shifted due to the change in domain. Therefore, only the denotations need to be adapted to a new domain model.

Acknowledgements

I am grateful to the guest editor and to an anonymous reviewer for their valuable suggestions on improving an earlier version of this paper.

Notes

1. We assume a terminological distinction made by Hovy and Nirenburg (1992): a language-*neutral* representation (as opposed to a language-*independent* one) can be mapped to any of the target languages the system handles, but does not claim to be independent of natural language as such.
2. Earlier NLG work has often neglected the need for re-structuring input representations; as an exception, Horacek (1990) argued the need for doing so.
3. Schmitt (1986) made the same point in his study of various technical manuals.
4. One could envisage using the whole subsumption machinery of LOOM for defining $C_1 \succeq C_2$, but in practice it seems sufficient to treat it merely as the transitive closure of the explicitly asserted subtype relationships in the DM, which can be determined by a simple traversal of the taxonomy (corresponding to Woods' (1991) "recorded subsumption"). It is, at any rate, important, to distinguish the basic $\succeq$ notion of subsumption from the *subsume* relationship we are defining here *in terms of* the former.
5. We do not address the problems of efficiently searching and matching lexical candidates here; they are discussed for instance by McDonald (1991).
6. For an extensive historical survey on the notion of connotation, see Garza-Cuarón (1991).
7. In NLG, collocations are treated, with different approaches, by Smadja and McKeown (1990) and Wanner (1994).
8. Also, Reiter (1990) discusses pragmatic reasons for not using the most specific word in certain situations of utterance.

References

Allerton, D.J.: 1982, *Valency and the English Verb.* London: Academic Press.
Bach, E.: 1986, 'The Algebra of Events', *Linguistics and Philosophy* **9**, 5–16.
Bateman, J., R. Kasper, J. Moore, R. Whitney: 1990, 'A General Organization of Knowledge for Natural Language Processing: The Penman Upper Model', Tech. Rep. USC Information Sciences Institute, Marina del Rey, CA.
Bateman, J., C. Matthiessen, K. Nanri, and L. Zeng: 1991, 'The Re-use of Linguistic Resources Across Languages in Multilingual Generation Components', in: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-91),* Sydney, pp. 966–971.
Bierwisch, M.: 1983, 'Semantische und Konzeptuelle Repräsentation Lexikalischer Einheiten', in: R. Ružička and W. Motsch (eds): *Untersuchungen zur Semantik.* Berlin: Akademie-Verlag, pp. 61–99.
Busemann, S.: 1993, 'A Holistic View of Lexical Choice', in: H. Horacek and M. Zock (eds): *New Concepts in Natural Language Generation.* London: Pinter, pp. 302–308.
DiMarco, C., G. Hirst, and M. Stede: 1993, 'The Semantic and Stylistic Differentiation of Synonyms and Near-Synonyms', in: *Proceedings of the AAAI-93 Spring Symposium on Building Lexicons for Machine Translation*, Stanford University, pp. 114–121.
Dorr, B. J.: 1993, 'Interlingual Machine Translation: A Parameterized Approach', *Artificial Intelligence* **63**, 492.
Dorr, B.J., C.R. Voss, E. Peterson, and M. Kiker: 1994, 'Concept-Based Lexical Selection', in: *Proceedings of the AAAI-94 Fall Symposium on Knowledge Representation for Natural Language Processing in Implemented Systems*, New Orleans.
Elhadad, M.: 1993, 'Using Argumentation to Control Lexical Choice: A Functional Unification Implementation', Ph.D. thesis, Dept. of Computer Science, Columbia University.
Garza-Cuarón, B.: 1991, *Connotations and Meaning.* Berlin/New York: Mouton de Gruyter.

Hawkins, J. A.: 1986, *A Comparative Typology of English and German.* London/Sydney: Croom Helm.
Helbig, G. and W. Schenkel: 1973, *Wörterbuch zur Valenz und Distribution deutscher Verben.* Leipzig: VEB Verlag Enzyklopädie.
Henschel, R.: 1993, 'Merging the English and the German Upper Model', Technical Report, GMD/Institut für integrierte Publikations- und Informationssysteme, Darmstadt.
Horacek, H.: 1990, 'The Architecture of a Generation Component in a Complete Natural Language Dialogue System', in: R. Dale, C. Mellish and M. Zock (eds): *Current Research in Natural Language Generation.* London: Academic Press, pp. 193–228.
Hovy, E.: 1988, *Generating Natural Language under Pragmatic Constraints.* Hillsdale, NJ: Lawrence Erlbaum.
Hovy, E. and S. Nirenburg: 1992, 'Approximating an Interlingua in a Principled Way', in: *Proceedings of the DARPA Speech and Natural Language Workshop.* Hawthorne, NY.
Iordanskaja, L., R. Kittredge, and A. Polguère: 1991, 'Lexical Selection and Paraphrase in a Meaning-Text Generation Model', in: C.L. Paris, W.R. Swartout and W.C. Mann (eds): *Natural Language Generation in Artificial Intelligence and Computational Linguistics.* Dordrecht: Kluwer Academic Publishers, pp. 293–312.
Jackendoff, R.: 1990, *Semantic Structures.* Cambridge, MA: MIT Press.
Jacobs, P. S.: 1987, 'Knowledge-Intensive Natural Language Generation', *Artificial Intelligence* **33**: 325–378.
Kasper, R.: 1989, 'A Flexible Interface for Linking Applications to Penman's Sentence Generator', in: *Proceedings of the DARPA Workshop on Speech and Natural Language Processing*, University of Pennsylvania, pp. 153–158.
Leech, G.: 1981, *Semantics.* Harmondsworth, UK: Penguin, (2nd edition).
MacGregor, R. and R. Bates: 1987, 'The Loom Knowledge Representation Language', Technical Report ISI/RS-87-188, USC Information Sciences Institute, Marina del Rey, CA.
Matthiessen, C. and J. Bateman: 1991, *Text Generation and Systemic-Functional Linguistics: Experiences from English and Japanese.* London: Pinter.
McDonald, D.D.: 1991, 'On the Proper Place of Words in the Generation Process', in: C.L. Paris, W.R. Swartout, and W.C. Mann (eds): *Natural Language Generation in Artificial Intelligence and Computational Linguistics.* Dordrecht: Kluwer Academic Publishers, pp. 227–248.
Miezitis, M.M.: 1988, 'Generating Lexical Options by Matching in a Knowledge Base', Technical Report CSRI-217, University of Toronto.
Moens, M. and M. Steedman: 1988, 'Temporal Ontology and Temporal Reference', *Computational Linguistics* **14**(2): 15–28.
Nirenburg, S. and I. Nirenburg: 1988, 'A Framework for Lexical Selection in Natural Language Generation', in: *Proceedings of the 12th International Conference on Computational Linguistics (COLING)*. Budapest, pp. 471–475.
Nogier, J.F. and M. Zock: 1992, 'Lexical Choice by Pattern Matching', *Knowledge Based Systems* **5**(3): 200–212.
Parsons, T.: 1990, *Events in the Semantics of English: A Study in Subatomic Semantics.* Cambridge, MA: MIT Press.
The Penman group: 1989, Unpublished Documentation of the Penman Sentence Generation System. USC Information Sciences Institute, Marina del Rey, CA.
Pustejovsky, J.: 1991a, 'The Syntax of Event Structure', *Cognition* **41**: 47–81.
Pustejovsky, J.: 1991b, 'The Generative Lexicon', *Computational Linguistics* **17**(4): 409–441.
Reiter, E.: 1990, 'Generating Descriptions that Exploit a User's Domain Knowledge', in: R. Dale, C. Mellish, and M. Zock (eds): *Current Research in Natural Language Generation.* London: Academic Press, pp. 257–286.
Rösner, D. and M. Stede: 1994, 'Generating Multilingual Documents from a Knowledge Base: The TECHDOC Project', in: *Proceedings of the 15th International Conference on Computational Linguistics (COLING)*, Kyoto, pp. 339–343.
Schmitt, P.: 1986, 'Die 'Eindeutigkeit' von Fachtexten: Bemerkungen zu einer Fiktion', in: M. Snell-Hornby (ed.): *Übersetzungswissenschaft – eine Neuorientierung.* Tübingen: Francke, pp. 252–282.

Smadja, F. and K. McKeown: 1990, 'Automatically Extracting and Representing Collocations for Language Generation', in: *Proceedings of the 28th Annual Meeting of the ACL*, Pittsburgh, pp. 252–259.

Somers, H.: 1987, *Valency and Case in Computational Linguistics*. Edinburgh: Edinburgh University Press.

Stede, M.: 1995, 'Lexicalization in Natural Language Generation: A Survey', *Artificial Intelligence Review* **8**: 309–336.

Stede, M.: 1996a, 'A Generative Perspective on Verbs and Their Readings', in: *Proceedings of the Eighth International Workshop on Natural Language Generation*. Brighton, pp. 141–150.

Stede, M.: 1996b, 'Lexical Semantics and Knowledge Representation in Multilingual Sentence Generation', Doctoral Dissertation, published as Technical Report CSRI-347, Dept. of Computer Science, University of Toronto.

Stede, M. and B. Grote: 1995, 'The Lexicon: Bridge between Language-neutral and Language-specific Representations', in: Working Notes of the IJCAI Workshop on Multilingual Text Generation. Montréal, pp. 129–135.

Talmy, L.: 1985, 'Lexicalization Patterns: Semantic Structure in Lexical Forms', in: T. Shopen (ed): *Language Typology and Syntactic Description 3: Grammatical Categories and the Lexicon*. Cambridge, UK: Cambridge University Press, pp. 57–149.

Tesnière, L.: 1959, *Eléments de Syntaxe Structurale*. Paris: Librairie C. Klincksieck.

Vendler, Z.: 1967, *Linguistics and Philosophy*. Ithaca, NY: Cornell University Press.

Wanner, L.: 1994, 'On Lexically Biased Discourse Organization in Text Generation', in: *Proceedings of the 15th International Conference on Computational Linguistics (COLING)*, Kyoto, pp. 369–378.

Woods, W.: 1991, 'Understanding Subsumption and Taxonomy: A Framework for Progress', in: J.F. Sowa (ed): *Principles of Semantic Networks*. San Mateo, CA: Morgan Kaufmann, pp. 45–94.

Wunderlich, D.: 1991, *Arbeitsbuch Semantik*. Königstein: Athenäum (2nd edition).