Lexical Knowledge for Natural Language Generation

Philip Edmonds
Department of Computer Science, University of Toronto
pedmonds@cs.toronto.edu

March 6, 1995

Abstract

In this paper, we examine how lexical knowledge has been modelled and used in generation systems of the past, and, looking to the future, what will be required of the lexicon to build versatile and expressive generators. Three central issues discussed are the following: the relationship between words and text structure, the connection between concepts and words, and the influence of context on word use and combination. We find that for a generator to be expressive, it must have knowledge of phrasal items, especially of collocations. We also find that the many available options for expressing a concept can be distinguished from another by pragmatic factors such as connotation, implicature, and style. Most importantly, this phrasal and pragmatic knowledge should be explicit and functional; it must prescribe how words are used, chosen, and combined.

"Every concept that can ever be needed, will be expressed by exactly *one* word,
with its meaning rigidly defined and all its subsidiary meanings
rubbed out and forgotten."

George Orwell, *Nineteen Eighty-Four*.

1 Introduction

Natural language generation is the computational process of producing text from an underlying 'conceptual' representation of that text. That is, it is the process of expressing some meaning in natural language. It requires knowledge of how to structure text, both rhetorically and syntactically, and knowledge of how words are used to express meaning. One of the central problems in generation is *lexical choice*. Words must be chosen that adequately convey the conceptual input of the generator, so lexical choice is, in one sense, the process of mapping concepts, whatever they look like, into real words.

Orwell's fictional prognosis was actually realized in many early text generation systems. But since then, researchers have endeavoured to build more complex systems that have required more of their lexicons. Today, text generation systems are sophisticated knowledge-driven processes. In this paper, we are concerned the building of versatile and expressive generators that can produce high-quality multi-sentential text. Obviously, such a system would have great demands on its lexicon, since words express most of the meaning of a text.

Usually a computational lexicon is thought of as just a repository of morphological and syntactic features of words. This idea undoubtedly comes from the long history of language analysis by computer. Another definition of a computational lexicon would have it be similar to a human-readable dictionary. But dictionaries and their electronic counterparts, machine-readable dictionaries (MRDs) and lexical knowledge bases (LKBs), are also unsuitable for text generation. While these traditional sources of lexical knowledge can contain a lot of sophisticated knowledge, they do not represent it in the right format. For generation, lexical knowledge must be made explicit because a generator must know how to use words—descriptive definitions and syntactic properties are not enough. A generator needs functional knowledge.

So, in this paper we will look at what sort of lexical knowledge is required by text generation systems, and how it should be represented. Lexical knowledge is simply knowledge about words. It has four main aspects: syntactic, semantic, pragmatic, and the purely lexical. The syntactic aspect includes morphological and syntactic properties that specify how words combine structurally. The semantic aspect is the most important because it specifies the meaning, or definition, of a word in terms of the concepts of some knowledge base. It specifies how words combine to express the meaning. The pragmatic aspect includes information about style, connotation, implicature, etc. Finally, some knowledge is purely lexical, not being syntactic, semantic, or pragmatic. Here we find idioms, inter-lexical constraints and lexical relations such as synonymy and hypernymy.

Two main problems that researchers focused on in the past are the relationship between words and text structure, and the relationship between words and concepts. In the first, we see that the choice of words and the choice of structure are processes that constrain each other. In the second, we see that one concept may map to several words, or map to none. A single word may also be ambiguous between several concepts. Section 2 of this paper explores solutions to these problems, in surveying the (recent) history of text generation.

A third problem that is now being addressed is the problem of how context affects word meaning and combinability. In sections 3 and 4 I will examine two avenues of recent lexical research that have sophisticated representations, both descriptive and functional, of how words combine. In section 3 we look at Pustejovsky's Generative Lexicon (Pustejovsky, 1991), and in section 4, Mel'čuk's Explanatory Combinatorial Dictionary (Mel'čuk and Polguère, 1987).

2 Lexical Knowledge in Generation Systems

Lexical knowledge can take many forms in natural language generation systems. In this section, I will discuss how lexical knowledge is organized and accessed by many of the systems that have been implemented in the past. The systems included below are intended to be a representative sample of the past research in generation.

2.1 Early Systems

One early sentence-level generator, MUMBLE, was developed by McDonald (1983; 1987). MUMBLE is able to map input propositions into phrase-sized chunks that are then grouped together, by means of planning, into appropriate surface structures. McDonald's main focus was the process of planning the syntactic structure, not the process of lexical choice; hence he assumed a very simple connection between semantic objects (MUMBLE's 'concepts') and words.[1] A semantic object such as **door-3** is mapped directly to the word *door*. So, only the most basic of lexical knowledge is used by the system: features relevant to morphology and syntax are stored in the lexicon.

One of the first complete text generation systems was McKeown's TEXT (McKeown, 1985). Lexical knowledge in TEXT is stored in several places because of the architecture of the system. Generation proceeds in two stages: first, according to the input, a text planner accesses a knowledge base of 'concepts' to fill up a schema, which is then passed to the second stage, the realizer.[2] The realizer takes the schema, and by using a functional unification grammar and a lexicon of morphological information, produces a surface utterance. The bulk of lexical knowledge is actually

[1]Generally in the literature, lexical entries are referred to as *lexical items* or *lexical units*. Cruse (1986) defines a *lexical unit* as participating in semantic/contextual relations, whereas a *lexeme* is just the orthographic representation of a word. However, in this discussion, I will simply call lexical entries *words*, although they could be phrases.

[2]This two-stage process (planning and then realization) has become a standard approach to the generation of texts larger than one sentence.

stored in a dictionary that serves as an interface between the two stages. The dictionary's main task is to select the appropriate word according to the syntactic role it must fill. For example, consider the semantic instance of **guided**, an object representing a *guided missile*: if an adjective is needed, it would translate to *guided*, and if a noun is needed, it would translate to *projectile*. The secondary role of the dictionary is to select, dependent on the choice of head verb, the 'deep structure', a specification of the syntactic objects and complements that the verb takes. Once the deep structure is chosen, the arguments of the deep structure are filled in. As in McDonald's system, McKeown's emphasis is on the structure (both rhetorical and syntactic) of the text, not on the choice of words. Although McKeown's dictionary is slightly more complex than McDonald's lexicon, each semantic object is directly linked to a group of words, distinguished only by syntactic features.

In these two systems and others not described here (see McDonald et al. (1987) for discussion on other early generators), there is a strict ordering of decisions. Lexical knowledge is not accessed until after planning the structure of the text, typically before or during syntactic realization. So, the actual choice of words is highly constrained by the already-chosen structure. This architecture led to the simple and direct mappings between concepts and words—a more complex mapping is not possible. The problem of being unable to find words to realize an utterance is thus avoided by the direct mapping; because once the concepts have been chosen and arranged, the words have already been selected.

But ironically, as Meteer (1992) points out, lexical choice is the most constraining task in a generator because words are used to express most of the meaning of an utterance. We will see that there are constraints on how words can combine, constraints on the arguments a predicate can take, and constraints resulting from the connection between concepts and words (in particular, there may not be a word to express a concept). So, a sufficiently complex generator that considers lexical knowledge only towards the end will seriously limit its expressiveness. It may simply limit the expressions available to choose from, or may even produce awkward or ungrammatical text.

Researchers soon realized that to improve the coverage and expressiveness of their systems, the systems would require a better command of lexical knowledge at all levels of generation. This meant that a text generator would have to at least allow for interaction between lexical and structural decisions. Researchers also realized that the problem of selecting words was much more important than was previously thought (Pustejovsky and Nirenburg, 1987); because, as application domains grew more complex, their lexicons also grew, and consequently concepts in the input could be realized by many possible words. This would inevitably require more complex mappings between concepts and words. In the two sections below, we first discuss the relationship between words and text structure, and then the mapping from concepts to words, although, as we can see, the two problems are interrelated.

2.2 Words and Text Structure

2.2.1 Traditional Systems

One solution to the problem of access to lexical knowledge is to not have an a priori order to decision-making in a generator at all. This solution was first considered by Danlos (1987) after she concluded that the conceptual, lexical, and syntactic aspects of generation are interdependent. In her system, two types of decision are made: those to select the simple sentences to express the principal concepts, and those to select a discourse structure. A lexicon-grammar specific to the domain is used in the first case, and a discourse grammar in the second. The priority of specific decisions can be determined by domain-specific heuristics. Despite this theory, her system was implemented in a few small and simple domains that did not require complex planning. The lexicon-grammars do not present significant word-choice problems beyond those related to syntax, and the discourse grammar is a table of possible constructions that depend on various constraints, especially those relating to causality and chronological order. Again, the concern here was making decisions about text structure, but some knowledge of how words interact with structure was taken into account.

A different solution, developed and implemented by Meteer (1992) in Spokesman, is to keep the standard decision-making order, but to allow lexical knowledge to be accessed from multiple places during generation. Spokesman is a two-stage system, including a goal-directed text planner built on top of McDonald's MUMBLE, that enforces a strict ordering on its decision-making. The text planner builds its plans out of a set of abstractions of linguistic knowledge (including lexical knowledge) that includes all and only that which is expressible; thus it builds plans that are guaranteed expressible. The abstractions consist of a sophisticated hierarchy of semantic categories and constituents that form the 'concepts' of the system. For example,[3] sentence (1a) is built from a number of basic constituents including *Peter* of abstract semantic category **named-object**, *made* of category **event**, *important* of category **property**, and *decision* of category **instance-of-a-kind**.

(1) a. Peter made an important decision to go to the beach.
    b. *Peter importantly decided to go to the beach.

The purpose of the semantic categories is to constrain the composition of constituents to produce only what is expressible. So, because the noun phrase constituent *a decision* is an **object**, an adjunct of category **property** is sanctioned in (1a); but in (1b) the **property** adjunct is not, because *decided* is an **event**. Contrast (1b) to (2), in which a **manner** adjunct, *quickly*, is allowed.

(2) Peter quickly decided to go to the beach.

Spokesman, in building a text plan, is able to access lexical knowledge (or abstractions of it), so any decision about structure can be constrained by the relevant portions of lexical knowledge.

[3] These are Meteer's examples, but Spokesman cannot account for them because its knowledge of semantic types is incomplete. It was not Meteer's goal to gather all such knowledge, which is obviously a very difficult task.

However, there is still a tight link between the basic constituents of text plans and words in the lexicon. The words *decision*, in (1a), and *decided*, in (1b), are directly linked to their respective constituents with no connection between them in the lexicon. A different word, like *resolved*, could not have been chosen instead. Since Meteer concentrated on the syntactic portion of linguistic knowledge, many constraints such as the one exhibited in sentence (1b) are not dealt with. The novelty of the system is the formalism that both represents expressible 'concepts' in an abstract hierarchical fashion and accounts for the use of this knowledge during generation.

2.2.2 Penman: A Systemic-Functional Generator

A less-traditional way of handling the interaction between syntactic and lexical knowledge is to treat the grammar and lexicon in the same unified manner. In the systemic-functional approach to linguistics (originated by Halliday (1985)), the lexicon is viewed as a simple extension of the grammar. A *systemic grammar* is a network of choice-points called systems, each responsible for choosing an appropriate grammatical feature. A traversal of the network causes choices to be made at each system encountered, resulting in a set of features that describes an utterance. The network is organized so that the more general features, those pertaining to syntax, are chosen before the more specific features, those pertaining to the lexicon. The set of features thus gathered can then be turned into a syntactic structure (such as a clause or a sentence) by using realization operators that are associated with the features. This functional approach is ideal for text generation since it can generate the appropriate structure, given a specification of the distinctions to be made in an utterance.[4]

Penman, a very comprehensive English language sentence generator, contains an implementation of a systemic grammar (of English) called Nigel (Penman Natural Language Group, 1988; Penman Natural Language Group, 1989). In Nigel, the most general systems choose grammatical features associated with structural and ordering operators. The more specific systems, encountered near the end of a traversal and thus constrained by the choices made before, choose features associated with realization (including lexicalization). Lexical items like *ship* and *en route*, in (3) below, are stored in a lexicon with only their morphological and syntactic properties. Each word is linked to a concept in an extensive linguistically-motivated knowledge base, or ontology. For example, one of the first and most general systems encountered chooses the **process-type** of a clause from the set of features {*material, mental, verbal, relational*}. So in generating (3), the feature *material* was chosen because the main action, an instance of the concept **en-route**, is subsumed by the ontological concept **material-process**. Later in the traversal, a very specific system is encountered in realizing the nominal group *the ship*. It chooses the feature *identifiable* (as opposed to *not-identifiable*), which is in part responsible for the realization of the definite article.

[4]The main thrust of this work, computationally, has been syntactic choice, but Matthiesen (1991) explains an abstract model for generation that includes a process called lexico-grammatical choice that combines lexical and syntactic choice.

(3) The ship is en route to Pearl Harbor.

The main problem with the systemic-functional approach is that lexical knowledge is not explicit. The success of Penman is a result of the surprising amount of implicit lexical knowledge encoded into the grammar and the knowledge base. Since the grammar and knowledge base were developed by a thorough investigation of English language usage, they take into account generalizations of pragmatic (stylistic and rhetorical), semantic, syntactic, and lexical knowledge. Although the knowledge base is extensive and deep, the lexicon itself is flat. The concepts in an input to the system may be organized into syntactic structures early in the process, but knowledge about words is not used until near the end. Lexical choice is simple because concepts like **en-route** map directly to words. The result is that word choice cannot constrain or influence syntactic choice.

An issue that comes to the forefront with systemic-functional generators is their dependence on the properties of a particular language. The lexicon, the grammar, and even the knowledge base in Penman are derived from a careful examination of English. While a great deal of lexical knowledge is inherently language-specific, the concepts to which words are linked should not necessarily depend on any language. The benefits of having language-independent knowledge are obvious for several types of text generator, including multilingual generators and those part of machine translation systems. So systemic-functional generators would not seem to be a good choice for such applications. However, this has not stopped the Pangloss machine translation project from pursuing an effort to merge the upper parts of Penman's knowledge base with other resources to form a language-independent ontology (Knight and Luk, 1994).

2.2.3 Phrasal Lexicons

In a sense, Penman represents the lexicon as part of the grammar. A different perspective would be to view the grammar as part of the lexicon, a view that starts by considering the phrasal patterns in language. Becker (1975) was the first to suggest that phrasal knowledge would be beneficial to natural language processing, if not a necessary component. He found that a large part of our linguistic knowledge is taken up in phrasal patterns, whether they are idioms such as *by and large* and *to kick the bucket*, phrasal constraints such as the one found in as *a heavy smoker* (to express "a smoker who smokes a lot", and not "a smoker who weighs a great deal"), or syntactic structures with spaces to be filled in such as ⟨Person A⟩ *gave* ⟨person B⟩ *a song and dance about* ⟨topic⟩. He estimates that "we probably know as many or more phrases than we know single words." This idea has been used in several implemented generators. Kukich's Ana (Kukich, 1988) uses the simplest notion of phrase by representing only frozen phrases in the lexicon. A 'macro-level' grammar has production rules that combine phrases to make sentences and paragraphs. However, the system and methodology are inadequate because the phrases are taken from a very restricted domain.

In Jacobs's PHRED, the phrasal lexicon is the central component of the system, consisting of pattern-concept pairs that make explicit links between phrasal patterns and semantic concepts (Jacobs, 1985). A phrasal pattern can be anything from a fixed idiom like

(4) $\langle$word = *the*$\rangle$ $\langle$word = *big*$\rangle$ $\langle$word = *apple*$\rangle$

to structural patterns like the following for the verb *remove*.

(5) $\langle$agent$\rangle$ $\langle$root = *remove*$\rangle$ $\langle$physical-object$\rangle$ $\langle\langle$word = *from*$\rangle$ $\langle$container$\rangle\rangle$

The concept associated with (4) is rather simple, but with (5) it is represented as a fairly complex proposition with variables that correspond to constituents in the pattern. The pattern-concept pairs encode a sophisticated combination of linguistic knowledge about conjugation (tense, infinitives, passivization), inclusion, ordering, and frozen or almost-frozen phrases. A matching algorithm determines which concepts, and thus patterns, should be used to express the input. By using a phrasal lexicon, syntactic and lexical decisions are unified under one mechanism, but, as in Penman, actual words are realized only towards the end of the process. One problem with PHRED is that the explicit one-to-one link between patterns and concepts is not conducive to complex lexical choice. A second problem is that choices are not well-motivated by factors beyond semantic and syntactic features.

Hovy's PAULINE is very similar to PHRED in its use of a phrasal lexicon, but it takes its use a step further because it incorporates rhetorical, pragmatic, and stylistic goals into the generation of larger texts. Hovy makes the strong claim that from the generator's point of view there is no need to make a distinction between grammar rules and lexical entries—they should be accessed in the same way. A lexical entry in PAULINE consists of either a syntax specialist that specifies complex functional knowledge, or a phrasal pattern. All decisions in PAULINE are motivated by both pragmatic and semantic factors. Apart from a standard semantic input, PAULINE takes a set of rhetorical goals that give rise to stylistic goals such as formality, simplicity, and haste. Word choice is governed by *discrimination networks*,[5] a form of decision tree, indexed by the stylistic goals and other factors. So to produce, for example, the more formal sentence (6a), the concept **death** is mapped to *died*, whereas in the colloquial (6b), an idiom is chosen instead.

(6) a. Janet died.
b. Janet kicked the bucket.

The major shortcoming of this work is, in fact, the use of discrimination nets. Since pragmatic information is encoded implicitly at the choice points of the net, it cannot be used by other planning stages of the generator. The fact that **death** can be realized as *die* or *kick the bucket* is not available to the planner so it cannot be used in planning the structure of the text (although once the phrase is chosen it will be added to the structure appropriately).

[5] A good survey on the use of discrimination nets in generation systems is given by Stede (forthcoming). We will discuss them further in section 2.3.1.

2.2.4 Collocations

A special and important type of phrasal pattern covered by Becker's phrasal lexicon, but not necessarily dealt with explicitly in the systems we have seen is called a *collocation*. According to Benson (1990), a collocation is an arbitrary and recurrent word combination. Although collocations can be viewed from many perspectives, the central idea is that they involve word co-occurrence that cannot be predicted from the semantics of the words themselves—they are exceptions to the normal rules of composition. But collocations are not entirely idiosyncratic either. For example, the word *rancid* will collocate with *butter* or *meat*, the combined meaning being composed of the parts, but not with *apple*, where *rotten apple* is used to express a similar meaning. Collocations are, however, idiosyncratic with respect to a particular language, so they are defined in terms of the lexical items of that language. There are so many collocations, for instance, in English that whole dictionaries, such as Benson et al.'s (1986), have been devoted to them.

For the purpose of text generation, collocations provide a major source of constraints for lexical choice. For example, from Smadja and McKeown (1991), to express the fact that *tea* has brewed for a long time, a generator must be constrained to choose the adjective *strong* over *powerful*. Or to express that a *suicide* has been attempted, the generator must choose *attempted* over *tried*. In Smadja and McKeown's representation, a collocation consists of a base, e.g., *tea* or *suicide*, that triggers the collocator, e.g., *strong* or *attempted*. They further organize collocations into syntactic categories such as adjective-noun and verb-noun exhibited here. The underlying principle in their generator, called Cook, is to lexicalize the words that will be the most productive in terms of collocational constraints: the bases of collocations. So, the arguments of the verb tend to get chosen first since the head nouns of noun phrases are usually bases. The verb is usually chosen last, in contrast to generators such as MUMBLE and TEXT, where the head verb is the first lexical choice made. In Cook the choice of a word influences the choice of other words in the sentence, and indeed, can influence the whole structure of the sentence.

In summary, there is a complex relationship between decisions about text structure, both rhetorical and syntactic, and decisions about word choice. The two types of decision can be viewed as distinct means of choice that constrain each other, or as just the two extremes of a decision-making process that accesses all forms of linguistic knowledge. The former is exhibited in the systems that allow for interaction between decisions by either representing lexical knowledge on several abstract levels (so it is accessible at different stages of generation), or by not ordering decisions a priori. The latter is found in the systems that use a phrasal lexicon as a central resource. Collocations and phrasal patterns are an important source of constraints for both structural and lexical decisions.

2.3 Mapping Concepts to Words

2.3.1 Semantic Factors

The first stab at a non–one-to-one mapping between concepts and words was the development of *discrimination networks* by Goldman (1975) in his BABEL generator. Instead of directly mapping to a single word, a discrimination net maps a concept to one of the the many 'near-synonymous' words that can express that concept by means of a static decision tree. Decisions usually involve semantic and lexical features but can also involve syntactic, and even pragmatic-like features, as we saw in Hovy's PAULINE. For example, after choosing the concept **ingest**, a discrimination net is traversed to determine if the concept should be realized as *eat* (if the object is solid), *drink* (if the object is alcohol), or *take* (if the object would improve the health), among others. But, in the case of BABEL, the conceptual primitives like **ingest** are so abstract that the task is rather silly—why should the input include **ingest**, when clearly **eat** or **drink** would be meant. While a discrimination net may seem like a good way to describe the fine-grained distinctions between words, it is difficult to motivate the decisions made at each node in the net. The distinctions between words are ad hoc and represented implicitly, so they cannot be used by other decision-making processes of the generator. A second problem, as elucidated by Danlos (1987), is that a the choice of a word over another often depends on factors scattered throughout the whole input semantic representation, not just on those local to the concept being expressed. This is to say that the context of the whole utterance is an important factor in determining word choice, but discrimination networks can make distinctions only on the basis of local factors.

The problem with many of the systems we have looked at so far is that they make no real distinction between words and concepts. The so-called concepts either map directly to words, or to opaque discrimination networks, so concepts and words are essentially the same thing. Lexical choice is hardly necessary, once the 'conceptual' content of an utterance has been chosen.

One of the first text generation systems to use a distinct representation of concepts and words was DIOGENES, developed by Nirenburg and his colleagues (Nirenburg and Nirenburg, 1988; Nirenburg, Lesser and Nyberg, 1989; Nirenburg et al., 1992). Building on the idea that decisions should not be ordered a priori, DIOGENES was given a central control structure called the 'blackboard'. In this system, many decision-making processes operate simultaneously, posting their results to the blackboard. A decision is triggered if its conditions have been satisfied by previous results. So lexical knowledge can be accessed, in theory, any time during generation, although the normal pattern is that the head of a clause is chosen first, followed by the other content words, and finally by the rest of the words (connectives, prepositions, etc.).

The architecture of DIOGENES also allows for significant improvements to lexical choice over the systems discussed above. Lexical knowledge is represented in a 'generation' lexicon (of words) that has links into a 'concept' lexicon, a taxonomic network of concepts, also called the domain model. Words are represented as frames that contain various features each with an importance factor ranging from 0.0 to 1.0. For example, *boy* has the feature **sex** with value male and importance 1.0, and **age** with value 2–15 years old and importance 0.4.[6] The frame also contains collocational information. For example, *boy* can be the agent of one of *play, throw,* and *jump*, and is associated with the places *school* and *playground*. Note how these collocations are organized around semantic categories like **agent-of** and **place**, which contrasts to Smadja and McKeown's organization around syntactic structures like noun-verb (i.e., subject-verb) or adjective-noun collocations. So, in DIOGENES, collocations can constrain the choice of words to fill semantic roles, like agent or location; whereas in Cook, they can influence only the choice for a particular syntactic role.

DIOGENES was one of the first systems, apart from PHRED, to treat collocational information explicitly in its generation process. Lexical choice in DIOGENES follows a complex matching algorithm that chooses between synonyms and near-synonyms, while obeying the collocational constraints that have arisen because of already-chosen words. Briefly, the algorithm first builds a candidate set of possible words to express the input concept. The set is then reduced by applying the collocational constraints. A *best match* is finally computed from the final set using a special distance metric that operates on the importance values. For example, the concept **young-male** could be realized by a word in the candidate set {*boy, kid, youth, teenager*}. But if *play* or *playground* had already been realized as part of an utterance, then a collocational constraint would require that *youth* or *teenager* be removed from contention. Finally, the distance metric would determine, in a context-insensitive way, whether *boy* or *kid* should be chosen.

Notice that in this system, an instance of a concept is not directly linked to a word or even a group of words, as in the previous systems. Instead, a matching process determines how a concept instance should be realized. But the problem with the approach is very similar to the problem with discrimination networks. Specifically, one concept maps to several words that are distinguished from one another by ad hoc features and importance values (see Stede (forthcoming) for details). Although the context of previously chosen words can affect word choice, a concept (with its features) will always map to the same word, since the distance metric is context-insensitive. In the above example, either *boy* or *kid* would be chosen depending solely on semantic features provided by the input. Not only are these features fairly arbitrary, but they do not represent pragmatic and stylistic dimensions. In the above, *boy* and *kid* are different stylistically at least in terms of formality.

2.3.2 Pragmatic and Stylistic Factors

The first system to use pragmatic and stylistic goals in generation was Hovy's landmark PAULINE. As we have seen, PAULINE chooses words according to stylistic goals such as formality, simplicity, and force. Words are chosen for their rhetorical effect. In addition, PAULINE has command of affect. Verbs, adverbs, and phrases can be assigned an affect that can be enhanced or mitigated as

[6] Importance values are assigned rather arbitrarily. In the case of describing a *boy*, his sex seems to be of greater importance to the description than his age.

phrases are constructed. For example, if the verb *tell* is seen as neutral, then its synonyms, *command* and *order*, are enhancers, while *ask* and *request* are mitigators. PAULINE is an important milestone in generation research because it highlights the fact that all decisions in a generator can and should be well-motivated by higher-level pragmatic, rhetorical, and stylistic goals.

In DIOGENES we saw that word choice can be constrained by collocational information. In DIONYSUS, a machine translation system developed in the spirit of DIOGENES, Nirenburg and his colleagues added *pragmatic constraints* (Onyshkevych and Nirenburg, 1992; Nirenburg et al., 1992; Nirenburg and Defrise, 1992). Pragmatic constraints pertain mainly to the style of a text, so they can influence both syntactic structure and lexical choice. To implement pragmatic constraints on lexical choice, lexical entries can be assigned 'attitudes', such as *evaluative* and *saliency*, that are given values ranging from 0.0 to 1.0. Pragmatic factors, a modified version of Hovy's stylistic goals, such as formality and colour, are also part of lexical entries and have similar ranges. Now, a more complex matching and decision process determines how to realize an input under the stylistic constraints imposed by the input. The input itself would be augmented with what Nirenburg et al.call the pragmatic profile of the speech situation, which includes information about the speaker and hearer, and the style of communication; but in DIONYSUS this information is kept to the minimum necessary to apply the pragmatic constraints.

The addition of collocational and pragmatic constraints to lexical selection is a great improvement over the earlier systems. The main problem, though, is that the mechanisms for choosing are still not well understood and so are too simplistic. Lexical representation is plagued by ad hoc decisions and simplifications made by the designers just to get the system running. For example, Nirenburg and Defrise (1992) admit that having only one simple evaluative scale ranging from *good* to *bad* is quite inadequate when trying to represent sentences containing words like *interesting*, *important*, and *boring*.

As part of the current TECHDOC Project (Rösner and Stede, 1994; Stede, 1993),[7] Stede summarizes the factors that should be represented in the lexicon of a versatile and expressive generator. First, the semantic specificity of near-synonyms requires fine-grained distinctions. For example, one must represent somehow the semantic distinctions between *see*, *look*, and *watch*. Second, collocational constraints must be accounted for, as we have seen in the DIOGENES system. Third, as in the DIONYSUS system, lexical style must be accounted for, but not as constraints on generation, but as *preferences*.

In contrast to DIONYSUS, however, Stede's work on representing stylistic features for words attempts to be more rigorous and systematic. He advocates validating his intuitions by psychological experiments. Again based on Hovy's work, Stede identifies seven stylistic dimensions including formality, euphemism, and slant. To determine the value of each stylistic dimension for a word, the

[7] The goal of the TECHDOC project is to design and implement a system that generates multilingual technical documents from a language-independent knowledge base.

word is compared, by hand, with its synonyms in a context that is 'minimal' to avoid the semantic influence of other words. In this way, stylistic criteria can be separated from semantic criteria, something that discrimination nets do not do.

Stede has extended Penman with his lexical choice algorithm that takes as input a semantic network and additional pragmatic parameters. Unlike other systems, it determines *sets* of words that correspond to *configurations* of semantic concepts in the input network by applying semantic constraints. Then the words that together express the complete network, obey collocational constraints, and are 'maximal' under a *stylistic preference function* are chosen.

The factors that affect lexical choice are still not well understood, especially those dealing with style and connotation. In PAULINE, DIONYSUS, and Stede's system, we see that stylistic features are not enough by themselves to be effective—there are two many of them and their interactions are complex. So to cope with this, a higher-level stylistic or pragmatic goal is specified in the input. In Stede's system, a goal such as **newspaper** style or **slang** style is a parameter to the lexical choice process. The goal is then converted to a set of stylistic values that are matched against the values of words in the lexicon. For example, from the same semantic network, the system produces (7a) in **newspaper** style, and (7b) in **slang** style. Currently the conversion is very simple, requiring more development. But it is unclear how this will be done without applying to even higher-level rhetorical goals and intentions, thus requiring a redesign of the generator as a kind of intelligent agent.

(7) a. Tom evicted the tenants, then he tore down the building.
b. Tom threw the tenants out, then he pulverized the shed.

Hirst (1995) takes the idea of stylistic dimensions and turns it around. Instead of defining a word in strictly positive terms, as Stede does, Hirst's approach is to define a word in terms of its *difference* from its near-synonyms. Near-synonyms may differ in many ways, sometimes several at once, including differences in denotation (e.g., *lie* vs. *fib*), emphasis (e.g., *foe* vs. *enemy*), implicature (e.g., *mislay* vs. *lose*), and formality (e.g., *drunk* vs. *pissed*), among others (DiMarco, Hirst and Stede, 1993). Hirst proposes that differences be represented as objects that can be reasoned about at both the level of concept and the level of word. So, he argues that the conceptual hierarchy should be relatively coarse-grained, representing language-independent concepts, while fine-grained distinctions should be made in the lexicon, which contains sets of near-synonyms linked to the conceptual hierarchy. For example, the concept **untrue-assertion** would be linked to the set of near-synonyms containing *lie, falsehood, untruth, fib,* and *misrepresentation* that is coupled with a set of differences among the words. Although this representation has advantages over the taxonomic hierarchies whose basis is positive information, it is less obvious how it can be used for lexical choice. The current proposal is that lexical entries incorporate something akin to 'usage notes' that explain the differences and suggest appropriate usage.

In summary, we would like text generators to be versatile and expressive in a variety of domains.

Among other things, this means giving them large lexicons and large conceptual knowledge bases. The central problem is then mapping from an input expressed in the language of concepts to an output text (in the language of the lexicon). The above systems are presented progressively from the first systems that used simple one-to-one mappings, to systems that explicitly mapped concepts to sets of words, and finally to systems that use matching techniques to compare, on several levels, concepts to words. In the most recent systems, constraints and preferences operate at lexical, semantic, and pragmatic levels. As we can see, the connection between concepts and words is complex. The fine-grained distinctions among words (of many types) must be represented explicitly to be useful in a generator.

2.4 Some Preliminary Conclusions

The main observations of this section are:

- Lexical knowledge should be accessible to all parts of the generation process. Different types of lexical knowledge may be accessed from different parts.
- Decisions of the generator should not be ordered a priori because of the interdependence of all linguistic knowledge.
- A generator requires a knowledge of idiosyncratic, collocational, and phrasal combinations of words.
- The mapping between concepts and words is not trivial. Lexical choice must match concepts to words using constraints and preferences on several levels of representation: lexical, syntactic, semantic, pragmatic, and stylistic.
- Decisions should be motivated by higher-level goals of the generation process. This means representing lexical knowledge that will help achieve these goals.

Generation research over the years has concentrated mainly on the first three points, and only recently have researchers focused on the last two points. The earlier systems discussed here do not really use words at all. Concepts in the knowledge base map fairly simplistically to lexical items, so the choice of words is actually done early on in the planning stage when the concepts to be expressed are chosen. Perhaps this is because the actual lexicons of the systems are relatively small and simple compared to the 'mental lexicons' of people, so there is just no need to make any complex decisions about what words to include once the concepts have been chosen. The primary purpose of lexical knowledge has been to constrain syntactic choice.

But now that we are looking towards building large lexicons, lexical choice has become an important and difficult problem to solve. Three of the central problems facing lexical choice that we have only alluded to here are:

1. the non-isomorphic connection between concepts and words: a single concept can map to many words, or a concept may have no word to express it at all;
2. the 'non-semantic' meaning component of words: the choice of a word can carry with it implicatures or connotations that are not accounted for by the concepts;
3. and the influence that context has on the meaning and choice of words.

We have already seen some examples of the first problem and possible solutions above; whereas the second problem is only beginning to get attention in work such as Hirst's (1995). For example, one difference between the semantically equivalent *mislay* and *lose* is that the former implicates an expectation that the item will be found and the latter that it won't. Lexical choice must account for such implicature, and other pragmatic factors, so that unwanted effects do not occur.

It should also come as no surprise that the linguistic context or situation an utterance is produced in plays a large role in the meaning of words and their combinations, and thus the choice of words. There are actually two separate notions of context to consider. First, the local context concerns the already chosen words in an utterance that can constrain the further choice of words in that utterance. The application of collocational constraints is one way to account for with this type of contextual effect. Second, the global context refers to the situation or environment in which a text is being produced. It includes what has been uttered already, the mental state of the speaker or writer, and the style of communication.

Of all the generators discussed above, only those that incorporate collocational and pragmatic knowledge (PAULINE, DIONYSUS, and TECHDOC) come close to allowing for contextual influence on word choice. But these systems do not go far enough because they do not address how the meaning of a word depends significantly on the other words that it is combined with and on the nature of the combination. For example, in (8) we see that *move* is ambiguous between a sense implying physical movement and a sense implying a psychological effect. The difference in meaning is largely a result of the verb's object, an inanimate object in (8a), and a sentient being in (8b).

(8) a. Please move your car.
b. Her pitiable story moved him deeply.

The standard way of dealing with lexical ambiguity has been to exhaustively list all word senses in separate lexical entries. But Sanfilippo et al. (1994) give two reasons why this *word sense enumeration* is at least inadequate from the perspective of language analysis.[8] First, it fails to make regularities in usage and sense extensibility explicit. For example, many verbs of caused motion (e.g., *agitate, crash, cross, lift, strike, sweep, unwind*) exhibit the same dual behaviour as *move* above. And second, it cannot account for the creative use of words. For instance, in (9), *VISA*

[8] A third reason they give is that word sense enumeration generates massive lexical ambiguity, an obvious problem in language interpretation.

is used in a novel way (although it is gaining acceptance) as an action verb, when it is normally thought of as a noun (being an object and a means of payment).

(9) I'll VISA the bill.

To start thinking about the contextual effects on word choice we need to realize that there are two sides to lexical knowledge: the *static* representation of features and relations, and the *functional* aspects that prescribe and explain usage. The regularity we see in word usage can be generalized and made explicit in functional knowledge. Both types of knowledge are necessary in order to have a fully prescriptive and explanatory account of lexical knowledge, but only the first has been dealt with seriously.

In light of this review, I will now discuss in depth two current approaches to modelling lexical knowledge that address its descriptive and functional sides, as well as word combinability and collocational behaviour. First, taking a lexical semantics perspective, some researchers have looked at the problem of context and word meaning. Central to this line of thought is the Generative Lexicon, in development by Pustejovsky and his colleagues. Second, in a linguistic and lexicographic approach, Mel'čuk's Explanatory Combinatorial Dictionary is a prescriptive representation of word usage and combinability.

3 The Generative Lexicon

3.1 Motivation

Pustejovsky and his colleagues have been developing the theory of the Generative Lexicon (Pustejovsky and Anick, 1988; Pustejovsky, 1991; Pustejovsky and Bouillon, 1994) for a number of years now. They had noticed several problems in lexical semantics that were difficult to solve in the standard feature-based or primitive-based formalisms. Consider the following three examples from Pustejovsky (1991).

(10) a. a fast typist
b. a fast car
c. a fast waltz

(11) a. Mary painted the door.
b. Mary walked through the door.

(12) a. John baked a cake.
b. John baked a potato.

In the first example, *fast* applies to different scales depending on the noun it is modifying. So then, should *fast* have multiple entries in the lexicon, one for each type of scale that it could refer to? This solution is not satisfactory because it does not take into account the similarity of each of its uses, least of all the difficulty (or the impossibility?) of listing all possible uses. A normal view of compositionality seems insufficient here.

In the familiar example (11), *door* appears to have two senses: in the first to refer to the physical object, and in the second to refer to the opening (assuming Mary is not a ghost). Again, the two senses could be placed in separate lexical entries, but this would miss the relationship between the two readings, possibly leading to strange interpretations of (13) (just what does *it* refer to?).

(13) Don't come through that door, *it* was just painted.

Since there are other nouns that exhibit similar behaviour, such as *window*, it would be advantageous to have a more generalizable formalism.

In example (12), the verb *bake* is ambiguous between a sense implying creation and a sense implying change of state. The standard view is that *bake* should have two lexical entries, because the verb takes two different types of complement. Standard views of lexical semantics put a lot of weight on the predicate/argument structure of a language; thus it is appropriate to posit many related senses of verbs (ignoring any structure for arguments) that are connected by transformations. But the two senses of *bake* are clearly related; thus we would like to represent this relationship so that the words can be used productively.

The standard models of lexical knowledge have a problem in representing the regularities we see in the above examples primarily because word senses are listed individually. So, the standard formalisms do not model how the meaning of a word can vary (in a logical way) depending on the words it is combined with, beyond the application of simple collocational constraints. In fact, these examples are collocations but they are not as idiosyncratic as the simple lexical collocations like *heavy smoker* and *rancid butter*, because they can be predicted from the semantics of the words involved. That is, they are more regular, but not entirely like free-combining words.

The methodology behind Pustejovsky's formalism is then that it both describe and explain actual language usage, and that it represent lexical entries not as a fixed and static set of semantic primitives, but in a flexible and generative fashion where word senses are related by logical operations. Furthermore, since nouns and adjectives exhibit behaviour just as complex as that of verbs, the semantic load that verbs usually take should be spread out over all open-class words.

3.2 Four Levels of Lexical Semantic Representation

In Generative Lexicon Theory, lexical meaning is characterized by at least four levels of representation. The first level, the *argument structure*, is a standard definition of how the semantics of a lexical item affects its syntactic expression. This level is primarily used to represent the semantics of verbs, so it is a minimal encoding of lexical semantics, at best.

The second level is the *event structure*, which represents the event-based semantics of words. Each lexical item can be associated with an event type, such as state, process, or transition, and as lexical items are combined to form larger expressions, their event types are also composed. For example, a process sequentially followed by a state may compose to form a transition.

The so-called *qualia structure* is perhaps the most innovative facet of the theory. Each nominal and adjective has a qualia structure which encodes the essential (and default) aspects or attributes of the word's meaning. These attributes are divided into four roles:

CONSTITUTIVE — the relation between the object and its constituents
FORMAL — that which distinguishes the object in a larger domain
TELIC — the purpose and function of the object
AGENTIVE — the factors involved in the origin or 'bringing about' of the object

The qualia structure is an abstract representation of lexical relations. It explains how words can be combined dependent on their semantics. The qualia structure is also a representation of collocational behaviour. We will explore this below in section 3.4.2.

Consider the following qualia structure for the noun *book* (taken from Pustejovsky et al. (1993)):

book(x,y):

CONSTITUTIVE: information(y)
FORMAL: physical-object(x)
TELIC: read(T,$reader$,y)
AGENTIVE: write(T,$writer$,y)

The variables x and y hold the two aspects of a book, seen in the FORMAL and CONSTITUTIVE roles. Two default actions, *reading* and *writing*, both transition events (marked by T), are associated with a book. The former is in some sense the purpose, or TELIC role, of a book; while the latter is what creates the information in a book, its AGENTIVE role. The two actions are encoded in separate roles either of which may be referred to depending the context of the word's use—specifically on other words used with *book*. For example, sentence (14a) emphasizes the CONSTITUTIVE role, (14b) emphasizes the FORMAL, and (14c) is ambiguous between a TELIC and AGENTIVE reading.

(14) a. This is an interesting book.
b. This book weighs four ounces.
c. Steven finished another book.

One may ask: why only four qualia roles? Pustejovsky feels that a minimal semantic description is determined by these four roles. Assuming one can categorize the semantics of a nominal into these four roles, there is still a problem. The roles are defined to contain only 'default' actions, that is, those actions most commonly associated with the object in question. But there are many actions that could potentially be associated with a nominal. In particular, the TELIC role of a nominal can be an enormous set. Consider the book: every different type of book-user would have his own purpose in using it such as reading, burning, propping (up a projector), ripping, etc. The fourth level of representation is intended to solve this problem.

The fourth level is the *lexical inheritance structure*. It encodes how lexical items are globally related to other items in the lexicon. The structure includes a *fixed* taxonomy: a standard semantic network that includes relations such as hyponymy and meronymy; and a dynamic inheritance mechanism called *projective inheritance* that can create a relational structure for arbitrary categories. The mechanism attempts to capture obvious commonsense relationships between lexical items by manipulating the values of qualia roles in specified ways. For example it would link *prisoner* and *escape*, but not *prisoner* and *eat*, because of the stronger intuitive relationship in the former. In this way, they claim, all of the TELIC roles of the above book could be determined. One potential problem with this method is the huge set of roles it can generate: how would one disambiguate sentence (14c)? One answer is to look at the context n which the sentence is uttered—we knew *Steven* to be a prolific writer, then perhaps he finished *writing* the book. Moreover, would not the 'default' roles of an object depend on the context? Well, the lexical inheritance structure is not fully developed as yet, so it is unclear that it will be able to solve this problem. For now, it gives many possible options without providing a mechanism for choosing among them.

3.3 Overall Organization

The multi-layered representation forms the static or descriptive portion of lexical knowledge. The procedural knowledge required to use the static knowledge is a fixed set of *generative devices* or lexical rules. In language analysis, the rules are used to construct semantic expressions for utterances by accessing all four levels of lexical representation, and in generation they are used to list all possible options available for realization.

One generative device in particular is *type coercion*. It is used to explain the examples (10) through (14) above. For example, the adjective *fast* is said to modify a process event in the TELIC role of an object. So the type of the modified noun is coerced, if possible, into a process event, which determines the meaning of the expression. Since a *car* has *drive* as its TELIC role, *a fast car* is one that is (or can be) driven quickly. The other examples follow similarly. Looking at (14c), the verb *finish* takes a transition event as complement, but it is actually complemented with a direct object. In the qualia structure for *book*, there are two transition events, so the semantic type of *book* can be coerced into the reading or the writing event depending on other contextual factors. Type coercion needs to be a complex device because we do not want it to be applicable always. We would not want it to apply to *a fast flower* or to *Steven finished the rock*.[9] So, type coercion is sanctioned by certain constraints arising from the qualia roles of the lexical item to be expressed, as we have seen, and also the argument and event structure that it is part of (Pustejovsky and Bouillon, 1994).

[9]While these two examples make sense in some contexts, they are outside the realm of what the qualia structure is intended to deal with. The actions performed with respect to the *flower* and the *rock* are not default, or normal, actions. Again we see the problem of context influencing 'default' behaviour.

The crux of the theory is that word senses represented in the lexicon are not static, but flexible. So, the sense of a word in its context arises generatively by composition with other word senses. Multiple lexical entries for related senses of a word are not necessary. In fact, logical lexical ambiguity (or logical polysemy) is eliminated from the lexicon. As well, the semantic load normally heaped onto verbs, in the form of subcategorization frames and predicate/argument structures, is reduced by spreading the knowledge out into the qualia structures of the nouns and adjectives, and by giving the lexicon a functional role on top of its descriptive role.

3.4 Applications of the Generative Lexicon

The theory has been applied mainly to applications in natural language understanding with successful results. Anick and Bergler (1992) explore the systematic violation of selectional restrictions (logical metonymy) by applying the theory to the interpretation of expressions involving music terms such as *perform* and *practice*, and reporting verbs such as *announce* and *claim*. For example, while *announce* selects for a human subject, sentences like *IBM announced third quarter losses* are acceptable and sometimes even preferred. Type coercion and metonymic shift are further developed by Pustejovsky et al. (1993). Here they apply the Generative Lexicon to the acquisition of lexical items from large corpora. They also use their notions of metonymy and polysemy to gather lexical presuppositions and preferences for verbs. Pustejovsky (1991) discusses the Generative Lexicon with respect to verb subcategorization frames.

Although the use of the Generative Lexicon has been successful in applications like these, it is intended to be useful for any kind of natural language processing. However, only recently has the theory been applied to natural language generation. I will briefly discuss two applications that make extensive use of the qualia structure and of type coercion.

3.4.1 Breaking Selectional Restrictions

People will often use words in novel contexts. In particular, a speaker will sometimes break a selectional restriction—that is, they will use a word of a semantic type different from that required by their current utterance yet still be understood. For example, in (15),[10] a geographical location is used instead of the required event (or time interval), which could have been expressed as *your vacation in Florida*.

(15) Would you like to visit us before or after *Florida*?

McDonald and Busa (1994) apply the Generative Lexicon in their model of language generation that accounts for when a selectional restriction can be broken.

Expressions like the above can be generated by representing the semantics of the words in the qualia structure. Type coercion is then used to model when selectional restrictions can be

[10] This example is a simplified version of one given by McDonald and Busa (1994).

broken. In this example, the lexicon would provide to the generator at least two options to express the desired meaning. It would first provide the selectionally correct expression *your vacation in Florida*, or perhaps *before you leave for your vacation in Florida or after you return*. And second, it would provide the simple *Florida*, since the TELIC role of *Florida* indicates that people vacation there, so its type can be coerced into an event (of vacationing).

3.4.2 Semantic Collocations

In their application of the Generative Lexicon, Viegas and Bouillon (1994) explore the problem of representing collocational information. A collocation is usually defined as an arbitrary and recurrent word combination. But sometimes words are combined in ways that are not entirely arbitrary, as we saw in examples (10), (11), and (12) above, in which the combinations depend in some way upon the semantics of the words. From their broad definition that "collocational behaviour emerges from a theory of what the range of connections and relations between lexical items can be," Viegas and Bouillon derive a range of collocational behaviour. At one end of the range we find fully compositional, or free-combining words, and at the opposite, completely non-compositional idioms. They define a type of expression somewhere in the middle of the range that is not fully compositional, but can be predicted from the semantics attributed to its lexical items, called *semantic collocations*.

The qualia structure of the Generative Lexicon can represent exactly this type of collocation. For example, assuming the standard mental state sense of *sad*, consider the following:

(16) a. a sad person
b. a sad book
c. *a sad tree

The first case is a normal use of the adjective to modify a noun of type human. In the second and third case, *sad* modifies an object, but only makes sense for the *book*. Why is there a difference? Viegas and Bouillon claim that (16b) has the normal reading 'a book that makes the reader sad', whereas (16c) has no analogous reading. So they conclude that the argument and qualia structures of *sad*, in its mental state sense, should specify that it can modify a noun that is of type human, or a noun that has some event in the TELIC or AGENTIVE role involving a human agent. Since a *book* has the TELIC role of *reading*, (16b) can be generated; but (16c) cannot because a *tree* has no TELIC or AGENTIVE roles involving humans since it is a natural object.

However, Viegas and Bouillon seem to have overlooked the reading 'a tree that makes its observer sad', or 'a tree that looks sad to its observer'. Perhaps this is because they focused on the coercion of the modified noun's type, thus ignoring the fact that *sad* is ambiguous between two senses, the second being a 'sad-looking' sense, in much the same way as *bake* is in example (12). If we consider both senses, then the meaning of *sad* can be shifted depending on the object it modifies. For (16c), the shift to the sad-looking sense is necessary because no other readings are available, but for (16a) and in (16b), we now find two ambiguous readings. The adjective can then coerce the FORMAL role as well as the TELIC and AGENTIVE roles of a noun.

In any case, an important observation here is that some adjectives act in regular ways. That is, they will modify nouns in similar underlying ways, while on the surface expressing different meanings. To make this formal, Viegas and Bouillon show that the argument and qualia structures of an adjective will specify what the adjective can modify. The possibilities include objects (of a specific semantic type), events (of a specific type such as transition, state, or process), and events in TELIC or AGENTIVE qualia roles. Then, the qualia structure of a noun contains the information that allows it to be combined with an appropriate adjective to achieve a desired reading.

One problem with this work is that one must distinguish between the semantic collocations of a language and the purely idiosyncratic ones. Clearly, the line between the two is fuzzy. When does a predictable combination of words become an idiosyncratic lexical co-occurrence? The designer of a lexicon must make ad hoc decisions about whether a particular collocation is general enough to be encoded implicitly in the qualia structure or explicitly as an idiom. Viegas and Boullion are unclear on this, and seem to imply a reverse definition of semantic collocation: that they are what results from any regularity in collocational behaviour that can be encoded in the Generative Lexicon.

But problems could then arise because of the range of 'regular' behaviour seen in adjectives. Consider the adjectives *big* and *heavy*, which can both modify the TELIC role of a word, giving us expressions such as *a heavy drinker* and a *big eater*. Without any other constraints, if the generator was asked to express 'a person who smokes a great deal,' it would be given several options by the lexicon including *a heavy smoker* and *a big smoker*. The former expression is appropriate, while the latter is not, largely because of idiosyncrasy. Further constraints are obviously necessary, accounting for the range of collocational behaviour between the semantic collocations and idioms. In section 4 we will look at precisely this type of collocation in the form of Mel'čuk's lexical functions. Here, Viegas and Bouillon have only begun to deal with these problems, so it remains to be seen whether this method will be adequate for large lexicons with many co-dependencies among words.

3.5 Conclusion

In essence, the Generative Lexicon is a typed-feature representation of lexical semantics. What makes it different from the more standard theories is its reliance on process as well as data. The sense of a word is not encoded as a static set of primitive features, but is built up by the so-called generative devices or lexical rules. This follows from the crucial observation that the meaning of a word actually arises from its combination with other words in a particular context—alone, a word has very little useful meaning.

The main contribution of Generative Lexicon Theory, whether or not its actual implementation is good, is that a formalism for lexical semantics should be generative, not exhaustive and static.

This is important for both the generation and interpretation of language, as the context is important to both processes.[11]

The second contribution of the theory, which is particularly important for NLG, is that the lexicon should both describe and explain word usage. In a highly expressive generator, the complexity of the lexicon would be huge, and thus require a sophisticated lexical choice process. With just a description of lexical behaviour, a large part of lexical knowledge is ignored making the task of lexical choice very difficult. With explanatory information about how and why words combine, in the form of the qualia structure coupled with generative devices here, a generator can then make principled decisions according to higher level reasons and goals.

The qualia structure is a novel formalism that is a powerful means of representing certain kinds collocational behaviour. In generation, we have seen this to be important for both using language creatively (McDonald and Busa, 1994), and for choosing appropriate combinations of words (Viegas and Bouillon, 1994).

One thing that the formalism is lacking is a lexical choice procedure. Clearly, one needs options to choose from, and the Generative Lexicon can provide many options, perhaps too many, but it does not distinguish between these options. Since the lexicon, developed in the tradition of lexical semantics, does not have an actual mechanism for choice, and it appears not to have been designed with this in mind, it could be a difficult task to design a mechanism that would interact with the lexicon appropriately.

Pustejovsky and his colleagues are motivated by the way words are used in real language, and their formalism reflects this. Now that the Generative Lexicon seems to be on firm ground, having many supporters, we should turn to analyzing its adequacy in interpretation and generation applications. The only way to do this is to build systems that use it, as has been done to show its adequacy in interpreting metonymy.

4 The Explanatory Combinatorial Dictionary

4.1 Meaning-Text Theory and the ECD

Mel'čuk and his colleagues have been developing a theory of language over the past two decades called the Meaning-Text Theory (MTT), where a Meaning-Text Model (MTM) is one particular implementation of the theory (Mel'čuk, 1981). MTT treats lexical knowledge as a primary source of linguistic knowledge, so its lexicon is a resource central to any model. The meaning component of the theory refers to only the linguistic meaning of an utterance in a particular language—thus it excludes any information to do with world knowledge—while the text component is simply the phonetic description of an utterance.

[11] Sanfilippo et al. (1994) and Ramsay (1994) give alternative formalisms that account for this behaviour, but only from the perspective of language analysis.

Briefly, the MTM describes the relation of a text to its linguistic meaning by representing an utterance on seven distinct levels ranging from a semantic level, through syntactic levels, to a phonetic level. An utterance can be mapped (bidirectionally) from its linguistic meaning to text through each level of representation by applying *paraphrasing rules* that replace part of an utterance with a new part at the same level, or *transformation rules* that convert the utterance from one level to another. Lexical knowledge governs the mapping process at each level, so the mapping rules that are applied depend on the lexical entries of the words involved.

Lexical knowledge is of course stored in the lexicon, called the *Explanatory Combinatorial Dictionary* (ECD) (Mel'čuk and Polguère, 1987; Mel'čuk and Zholkovsky, 1988). As is expected from the design philosophy that lexical knowledge plays a central role in language processing, the ECD is a unique and powerful resource. Specifically, the ECD of a particular language attempts to cover all possible *linguistic* knowledge, as opposed to world knowledge, relevant to that language. Mel'čuk calls the ECD an *active*[12] dictionary because its objective is to "give the user as complete a set as possible of the correct means for the linguistic expression of a desired idea (Mel'čuk and Zholkovsky, 1988, p. 44)." The ECD is also intended to be as explicit and systematic as possible in the representation of these means. To this end, it prescribes usage by explicitly relating every word to every other word that it can combine with in normal situations.

In the ECD, each 'major' word sense has its own entry. Although each entry contains all the information that a standard lexicon contains, such as part of speech and morphology, we are concerned here with only three of its structures: the semantic, the syntactic, and the lexical combinatoric structures.[13]

The semantic structure defines a word sense as a semantic network, the nodes of which represent other 'simpler' word senses in the dictionary. So, a word's definition is an intuitive paraphrase of its meaning that does not rely on semantic primitives. The definition is the most important part of an ECD entry, because the rest of the knowledge associated with a word sense is described contingent on its definition. Linguistic meaning, not syntax, is the motivating factor in this lexicon.

The syntactic structure of a word sense consists of, in part, the so-called government pattern that describes how the semantic arguments, variables in the semantic network, are realized as syntactic arguments. Perhaps an important point about the grammar should be made here. As we saw in section 2.2, a lexicon is usually treated as a kind of appendix to the grammar. In contrast, the ECD actually treats the grammar as an appendix to itself. The lexicon is the foundation of the MTM, with grammar in the form of paraphrasing and transformation rules that provide useful generalizations and procedures for manipulating lexical data.

Finally, the most innovative part of the ECD is the lexical combinatoric structure. It is a type

[12]In this terminology, the standard type of dictionary is *passive* because its purpose is to aid in the comprehension of texts.

[13]An ECD can be developed in a number of ways, including both human-readable formats, such as the ECD of French (Mel'čuk et al., 1984), and formats for computation, to be discussed here.

of dependency structure between word senses that systematically encodes the restricted lexical co-occurrence of every word sense. It is here that the vast number of *lexical collocations* are stored. A lexical collocation is a type of collocation that specifies a restriction on lexical co-occurrence that is idiosyncratic with respect to the real-world meaning of the word. This kind of restriction can be illustrated best by considering a phrase that has the same meaning but different lexical realizations over several languages. In (17), from Mel'čuk and Polguère (1987, p. 271), the verb cannot be selected according to the meaning of *question*. A different verb is used for each language presented: *ask* (or *put* or *pose*) in English, *put* in French, and so on.

(17)
a. English: *ask* a question
b. French: *poser* (lit. 'put') une question
c. Spanish: *hacer* ('make') una pregunta
d. Russian: *zadat'* ('give') vopros

4.2 Lexical Functions

Restricted lexical co-occurrence is encoded in *lexical functions*. Formally, a lexical function, **F**, is a mapping from one word sense to another word sense (or set of word senses) that expresses an abstract meaning and plays a specific syntactic role. Each lexical function expresses a certain kind of language-independent relation. The idea is innovative because it serves to generalize lexical collocation from being language-dependent to being language-independent. The set of lexical functions (with values) associated with a word embody the full range of collocational behaviour that the word exhibits; however, the set still needs to be listed exhaustively. Lexical functions also allow for systematic access to the collocations.

For example, the $\mathbf{Oper_1}$ function maps a noun N to the verb that takes N as its direct object.[14] Thus, $\mathbf{Oper_1}$ encodes the following restriction on *question*, for each phrase in (17):

(18)
a. English: $\mathbf{Oper_1}(\mathit{question}) = \mathit{ask}$
b. French: $\mathbf{Oper_1}(\mathit{question}) = \mathit{poser}$
c. Spanish: $\mathbf{Oper_1}(\mathit{pregunta}) = \mathit{hacer}$
d. Russian: $\mathbf{Oper_1}(\mathit{vopros}) = \mathit{zadat'}$

Mel'čuk defines about 60 *simple* lexical functions, including those in figure 1. Two lexical functions can also be combined to make a *compound* lexical function whose meaning is a 'combination' of the meanings of the individual function. Thus, $\mathbf{IncepOper_1}$ would mean 'begin performing':

(19) $\mathbf{IncepOper_1}(\mathit{contact}) = \mathit{come\ [in\ contact\ with]}$

[14] The subscript **1** in $\mathbf{Oper_1}$ means that $\mathbf{Oper_1}(N)$ takes the first semantic argument of N as its subject. Other semantic arguments can be specified with different subscripts, thus $\mathbf{Oper_2}(\mathit{question}) = \mathit{answer}$. This mechanism, common to many of the lexical functions, is one means of mapping the semantic network to syntactic representation.

$\mathbf{Oper_1}$: 'perform'	$\mathbf{Oper_1}$(*decision*) = *make*
Magn: 'intense(ly)'	**Magn**(*wounded*) = *badly*
	Magn(*decide*) = *important*
	$\mathbf{Magn^{temporal}}$(*decide*) = *quickly*
Syn: 'synonym'	**Syn**(*shoot*) = *fire*
Gener: 'hypernym'	**Gener**(*(to) lie*) = *deceive*
Anti: 'antonym'	**Anti**(*victory*) = *defeat*
$\mathbf{S_0}$: 'situation'	$\mathbf{S_0}$(*decide*) = *decision*
$\mathbf{S_1}$: 'situation actor'	$\mathbf{S_1}$(*shoot*) = *marksman*
$\mathbf{V_0}$: 'action'	$\mathbf{V_0}$(*decision*) = *decide*
Incep: 'begin'	**Incep**(*(to) sleep*) = *fall asleep*

Figure 1: A few simple lexical functions with examples.

The claim is that with only 60 lexical functions, they and their combinations can be used to describe systematically and exhaustively almost the whole range of restricted lexical co-occurrence in any language. However, a few problems can be identified.

Viegas and Bouillon (1994) find two problems. First, the set of lexical functions for a word sense is a static representation of its collocational behaviour. But, as we saw in section 3.4.2, some collocations are not entirely idiosyncratic and can be predicted from the meaning of the words involved. Second, the meaning expressed by a lexical function can be too general. For instance, a noun can have many dimensions along which to apply the **Magn** function. In particular, *experience* has at least two such dimensions: quantitative and temporal, as shown in (20). This solution is not desirable because we have to come up with new lexical functions for every new dimension we discover, which defeats the purpose of having a small, fixed, and relatively systematic set of functions.

(20) a. $\mathbf{Magn^{quantitative}}$(*experience*) = *extensive*
b. $\mathbf{Magn^{temporal}}$(*experience*) = *lengthy*

So in one sense, lexical functions are too general because they fail to express fine-grained semantic relationships; and in another sense, not general enough because they fail to represent predictable relationships. In both cases, it would be nice if lexical functions were more sensitive to the non-linguistic meaning of the words involved, but this is not possible because only linguistic meaning is represented in the MTM. Lexical functions carry an abstract kind of meaning, but only in an ill-defined implicit way. Despite these problems, lexical functions are a valuable way of representing idiosyncratic lexical collocations, whereas semantic collocations, discussed by Viegas and Bouillon, are better represented in a formalism such as the Generative Lexicon.

4.3 Two Applications of Lexical Functions

Because of the vast amount of lexical behaviour they can encode, lexical functions have been studied and applied by many researchers. We will look at two applications in the following sections. The first, called GOSSiP, is a sentence generator based on the MTM that is capable of paraphrasing an utterance in several ways. The second application is made outside the framework of the MTM. Here, lexical functions serve as the basis of a set of lexical resources for a complex lexical choice process in a text generator.

4.3.1 An Application Within the MTM

GOSSiP (Iordanskaja, Kittredge and Polguère, 1991) is a sentence-generation system that implements the MTM. The strength of the system is that it can generate a large variety of different lexicalizations from a single input by paraphrasing the utterance at its various levels of representation.

One source of paraphrase occurs in manipulating the deep syntactic structure by using lexical functions. A portion of the representation may be replaced by a new portion as determined by a *paraphrasing* rule. For example, rule (21) maps a verb node X to a syntax tree containing a semantically empty verb linked to a nominal.

(21) $X \Longleftrightarrow \mathbf{Oper_1}(\mathbf{S_0}(X)) \longrightarrow \mathbf{S_0}(X)$

By applying this rule to the deep syntactic representation of (22a), we get (22b). Notice that it is an entirely structural change leaving the meaning unchanged.

(22) a. Peter decided to go to the beach.
b. Peter made a decision to go to the beach.

Mel'čuk claims that about 60 paraphrasing rules are all that are necessary to cover all the systematic paraphrases of this nature in any language: they are language independent, just as the 60 lexical functions are. This mechanism is very powerful because, first, it allows a single input representation to be mapped to many possible sentences, and second, it provides 'backup' options if an initial representation cannot be expressed.

A second source of paraphrase occurs during the translation of the utterance from deep syntactic structure to surface syntactic structure. It is here that the lexical functions are finally mapped to their values. For example, if the verb *decided* in sentence (22a) had had a $\mathbf{Magn}^{\mathbf{temporal}}$ function as an attribute (represented as a dependent node in the syntax tree), then it would have been evaluated to *quickly* at this point, resulting in sentence (23a).

(23) a. Peter *quickly* decided to go to the beach.
b. Peter made a *quick* decision to go to the beach.

The significant thing to observe, though, is that since lexical functions are not evaluated until this point, the lexical consequences of earlier paraphrasing need not be considered until now. For example, if rule (21) is applied to the deep syntactic representation of (23a), the $\mathbf{Magn}^{\mathbf{temporal}}$ node would become a dependent node of the $\mathbf{S_0}(X)$ node. As shown in (23b), it is evaluated to *quick* instead of *quickly*, since $\mathbf{Magn}^{\mathbf{temporal}}(\textit{decision}) = \textit{quick}$.

Now, if *decided* in (23a) had been modified by a **Magn** function instead, then the sentence would not be expressible since **Magn**(*decide*) has no value. In this case, a paraphrase is necessary and would result in evaluating **Magn**(*decision*) to produce *important* (instead of *quick* in (23b)). Compare this implementation to Meteer's. In fact, these are Meteer's examples, sentences (1) and (2) on page 5, recast in Mel'čuk's formalism. Both systems use an abstraction of lexical knowledge, but Meteer's applies it during text planning, while Mel'čuk's during sentence realization (or planning). This difference results in the possibility that GOSSiP may have to backtrack because of paraphrasing a syntactic structure into something that is not expressible (because a lexical function does not exist).

As we discussed in section 2.2, the lexical resources available to express the meaning of a text are intimately related to the desired structure of that text. The MTM, and GOSSiP, solve one facet of this problem: lexical functions provide the bridge between words and structure. Since the paraphrasing mechanism changes the structure of a text, it must pay close attention to its lexical consequences, which it can do easily because all the consequences of structural changes are carried along abstractly in lexical functions, which are not realized until after the structure has been settled on. In other words, after the structure of a text is manipulated, the correct lexical choices will be made because lexical information is carried not in words, but in the lexical functions of words.

4.3.2 An Application Outside of the MTM

In his model of text generation, Wanner (1994a; 1994b) has made a substantial use of lexical functions. Where the MTM dealt with sentence-sized utterances, and thus with only the realization stage of a text generation system, Wanner's model is more comprehensive. It is a standard two-stage model in which the first stage builds a text plan that is then realized by the second stage.

The interface between these two stages has been a very contentious issue in text generation circles. Wanner's solution is to model the interface as a complex lexical choice process that can transform a text plan, built out of Mann and Thompson's RST relations (Mann and Thompson, 1987), into a grammatical structure that is given to a sentence generator.[15]

The lexical choice process makes use of a situation-specific, multi-layered model of lexical resources that is grounded in lexical functions, and ultimately words. As we will see, the benefits of using lexical functions in this model are similar to those in the MTM: lexical decisions can be

[15]The model is implemented in the framework of KOMET (Teich, 1992), the systemic (German) grammar based on Nigel. The output of the text planner is transformed into a *Partial Grammatical Structure*, a typed-feature structure, that is input to KOMET.

made alongside structural decisions, each constraining the other, while making sure that the utterance remains expressible. Lexical choice is thus integrated into the planning of both rhetorical and syntactic structure.

One problem with Mann and Thompson's RST relations is that they do not provide enough information by themselves to choose among the many lexical and syntactic means of realizing them. The problem is especially evident when related discourse segments contain information about the same topic: lexical constraints between the segments arise and must be met. For example, the sentences in (24) are all ways of realizing a **restatement** relation that differ in the lexical means employed, but the relation itself does not represent these finer-grained distinctions (rhetorical or otherwise).

(24) a. He is lying; he isn't telling the truth.
b. He is lying; he is deceiving me.
c. He is lying; he is telling falsehoods.

Wanner set out to taxonomize the lexical means of expressing discourse structure relations. He defines each of these means as a different *lexically biased discourse structure relation*, and organizes them into a taxonomy according to their lexical and semantic properties. Broad distinctions between relations are made at the top of the taxonomy and fine-grained ones near the bottom. So, for example, the **elaboration** relation subsumes the **restatement** relation, which, in turn, subsumes the four relations **contrastive-restatement** (seen in (24a) above), **generalizing-restatement** (in (24b)), **strict-restatement** and **intensifying-restatement**. **Strict-restatement** further subsumes **clarifying restatement** (in (24c)), among others. The taxonomy is encoded as a systemic network where each system represents a specific lexically biased discourse relation. An RST relation is mapped to a lexically biased discourse relation by choosing an appropriate entry point into the network, following which a traversal is made that depends on the functional, semantic, and syntactic features desired. Finally a particular relation is chosen, but this is only the first step in the lexical choice process.

Lexical choice is actually a multi-layered process that relies on four layers of lexical knowledge, each for describing increasingly smaller pieces of an utterance. The above taxonomy forms the intersituational layer, which describes how to express the relations between the predicates of an utterance, but not the predicates themselves. It is followed by the intrasituational layer, for realizing predicates and their attributes; the constituent layer, for realizing the constituents of predicates; and the lexeme layer, for realizing words. Lexical resources in each layer are encoded in systemic networks according to their functional, semantic, and syntactic features. The leaves of a network consist of chunks of *Partial Grammatical Structure* with links to networks in other (usually lower) layers. The choice process successively traverses networks in each layer starting from the top layer, and thus builds up a Partial Grammatical Structure that will be input to the sentence/clause generator (KOMET).

Now that we've seen the structure of lexical resources in Wanner's system, we can look at how lexical functions are used to encode the resources in each layer. The organization of lexical resources is situation-specific. That is, lexical resources are clustered together for each situation that is defined.[16] This means that a different set of networks are traversed to realize an utterance in, for instance, a LYING (the telling of falsehoods) situation, than in a SHOOTING situation. Wanner found this necessary because of the different functional, semantic, and syntactic properties of the lexical resources required to realize utterances in each situation.

Now, lexical functions correspond to different abstractions of lexical knowledge on several levels so they can be used in each of the four layers. At the lowest layer, the lexeme layer, lexical functions are associated with their values. This is identical to the lexical combinatoric structure for an entry in the ECD, the entry in this case being the situation.

Moving up to the the constituent layer, lexical functions are used to specify the ways of realizing the arguments and attributes of a particular situation. For example, the actor of a SHOOTING situation could be realized by using the situation-specific and neutral *marksman*, or by using the negatively-loaded *killer*, among others. These choices are specified in the actor-realization network as lexical functions. So, instead of finding the word *marksman*, we find the lexical function $\mathbf{S_1}$. A similar construction is used for *killer*.

While the use of lexical functions in the constituent layer is fairly simple and straightforward, their use in the intrasituational layer is quite interesting and innovative. The intrasituational provides an encoding very similar to the *paraphrasing* rules we saw in GOSSiP; but instead of paraphrasing syntactic structure, the appropriate structure is chosen by traversing the network. One problem with the paraphrasing rules was that there was no mechanism to dictate when a rule should be applied. Here, the problem is solved, since the criteria for choosing individual structures are explicitly encoded into the network. So, for example, in the actor-performance network, the feature *action-oriented* would result in using the function $\mathbf{V_0}$ to realize the action, giving utterance (25a), while the feature *situation-oriented* would result in using $\mathbf{Oper_1}$ followed by $\mathbf{S_0}$ giving utterance (25b). This portion of the network provides the same functionality as paraphrase rule (21).

(25) a. The marksman shot at Phil.
b. The marksman took a shot at Phil.

We can now see one problem with encoding lexical resources specific to situations: we lose the generality of the paraphrasing rules. While Mel'čuk found 60 language-independent, and obviously situation-independent, rules that cover all paraphrases of this type, Wanner must encode the rules separately in each situation, thus losing their generality.

[16] This is not unlike Danlos's (1987) domain-specific lexicon-grammars, which are defined individually for each domain. However, Wanner goes further and includes discourse structure as well.

Finally, moving to the intersituational layer, Wanner develops a theory of lexical function *sequences*. An LF sequence consists of two or more lexical functions in a specific order that all have the same argument. Lexically biased discourse relations are realized by using LF sequences. So the relation **contrastive-restatement**, illustrated in (24a), could be realized with the LF sequence $\mathbf{V_0}$,*not*+$\mathbf{AntiV_0}$; **generalizing-restatement** in (24b) with the sequence $\mathbf{V_0}$,$\mathbf{GenerV_0}$; and **clarifying-restatement** in (24c) with $\mathbf{V_0}$,$\mathbf{SynV_0}$.[17] Unfortunately, the theory becomes obscure at this point because LF sequences never explicitly appear in the representation of the layer. They seem to be used as a guideline for developing or compiling the contents of the networks in the lower layers. They do, however, explicitly represent the lexical constraints between related discourse segments.

The strong point of Wanner's system is the substantial lexical choice process that relies heavily on its multi-layered organization of lexical functions. Instead of directly choosing words, abstractions of words (lexical functions) are chosen, which are eventually evaluated to specific words. The power of this process is that as the structure of a text is built, all the available lexical resources are taken into account, providing many possible realizations without over-constraining the process (because of the choice of a particular word). So, in one sense, lexical and structural decisions are made simultaneously, thus allowing their constraints to interact.

While Wanner's encoding of lexical knowledge is substantially different from Mel'čuk's MTM, it subsumes the same functionality. It surpasses the MTM by providing an additional abstraction of lexical resources, the multi-layered model, and by providing a mechanism for choosing among possibilities. Where the MTM did not go beyond linguistic knowledge, Wanner's system does. Wanner's system integrates lexical choice with text planning, where MTM only models sentence-planning.

The major deficiency of the system results from the lexical resources being clustered around specific situations. Obviously this is necessary at the two lowest layers that are inherently dependent on the situation and the language, but at the two highest levels the argument is not so clear. It may be an important method of partitioning lexical resources for a working system, but it fails to represent the regularity seen in the application of lexical functions across situations. So, although the meaning of words would depend on the situation they are used in, it would be difficult to account for contextual influences from outside of a situation. By the same token, what constitutes a situation?

4.4 Discussion

The design philosophy that lexical knowledge plays a central role in language processing is important: it places the lexicon at the centre of a text generator so that all decisions depend on it. Both the ECD and Pustejovsky's Generative Lexicon (GL) observe this philosophy. In comparing the

[17]Refer back to figure 1 on page 26 for the description of these functions.

two models, we find that the GL represents lexical knowledge in a layer above the ECD. Since the ECD is a model of purely linguistic knowledge, it contains idiosyncratic language-specific lexical knowledge. Its lexical functions are used to encode lexical co-occurrence statically and exhaustively, which is beneficial for the truly arbitrary collocations, but not for others. On the other hand, the GL, in modelling some real-world knowledge, contains more general and predictable collocations. In this way we can see two models, not at odds with each other, but as part of a larger model of linguistic behaviour.

Both models have sets of rules for applying the knowledge in the lexicon: the GL has generative rules, and the ECD has paraphrasing rules. But where they differ is in the application of these rules. The GL, in its current form, does not specify when and how rules should be applied in language generation—that part of the model has not been designed yet. On the other hand, the MTM fully prescribes how (but not necessarily when) rules should be applied.

The MTM and its implementation in GOSSiP is very good at producing simple sentences because the input, a semantic network, can represent a sentence-sized utterance without too many problems. In scaling up to larger text, it is not clear that a semantic network can adequately represent that text; indeed, even in GOSSiP, an additional discourse structure had to be layered over the semantic structure to constrain the generation process.

An interesting comparison to make is between Penman, discussed in section 2.2, and the MTM, since both models were designed by linguists and lexicographers, not by the usual computational linguists. Both models are designed to generate sentence-sized output. In Penman, lexical knowledge is implicitly encoded into the grammar (Nigel), and the lexicon itself is a rather flat and simple construction containing syntactic features for words. The opposite is true of the MTM, where the lexicon is the central structure that controls the application of grammar rules. The difference is evident in the function of the systems: Penman has many explicit features concerning the structure of the output text, whereas the MTM has many explicit options for lexical choice. However, both systems do not account for the generation of texts larger than a sentence.

The problem arises in the MTM because only linguistic meaning is modelled, not 'real-world meaning'. Mel'čuk acknowledges this issue by asserting that, ultimately, two levels of representation of linguistic behaviour are necessary. At one level, we model the relationship between real-world knowledge and linguistic knowledge (a reality-meaning model), and at the second, we have the MTM. But Meteer (1992) has found that a two-level model such as this essentially separates text planning and text realization, which means that linguistic constraints (from the MTM) are not accessible to the planner. Meteer and others maintain that there should be a complex connection between these two levels, so that text plans are expressible by the realizer.

Wanner's system is an answer to this. It merges the explicit method of choice found in systemic formalisms with the complex lexical knowledge found in lexical functions. While its encoding of lexical knowledge is substantially different from Mel'čuk's MTM, it subsumes the same functionality. It surpasses the MTM by providing an additional abstraction of lexical resources, the multi-layered model, and by providing a mechanism for choosing among possibilities, so that larger texts can be generated.

Finally, on the concept-word front, there is very little development here. Words are defined by a purely linguistic meaning in terms of only other words. This makes word meaning intuitive and implicit, and excludes explicit factors from style, pragmatics, rhetoric, lexical semantics and context. Although lexical functions and paraphrasing rules, or their equivalent in Wanner's system, provide many options for a lexical choice process, without any other means of distinguishing between word senses, it is difficult to choose among the options.

5 Conclusion

To design a versatile and expressive generator we will need a large and complex lexicon. It must contain more than words, syntactic properties, and definitions. The building blocks of the lexicon should be idioms, words, and phrasal patterns associated with information pertaining to syntax, semantics, and pragmatics. However, one of the most important requirements of the lexicon is the functional knowledge of how words are used and combined.

In Mel'čuk's words, the lexicon should be *active*. That is, it must provide explicit and systematic knowledge of how words are used to express concepts. How is a generator to be expressive without this kind of knowledge? So, taking this perspective means that the lexicon should be regarded as a component central to a text generation system. This is not to say that it must drive the process, but it must be accessible during all activities of generation from the planning of rhetorical structure, to the planning of syntactic structure, and to the choice of words.

We have seen several recent systems that use a lexicon in this way. Meteer's Spokesman, Mel'čuk's MTM, and Wanner's system all represent lexical knowledge in the form of abstractions that are used during the planning of text structure. Pustejovsky's Generative Lexicon is also meant to be a central resource, but as yet it has not been embedded in an implemented system. While Meteer's system accounts for mostly syntactic applications of lexical knowledge (that is, how lexical knowledge constrains the text structure), ignoring for the most part lexical choice, the other three formalisms model how words are combined to express meaning.

The stress in these three systems is on representing *collocations*, one form of phrasal knowledge. As we have seen, the term "collocation" takes in a lot of behaviour, but collocations can be grouped loosely into types and organized in a hierarchy. In earlier systems, collocations were listed exhaustively, but categorized according to either their syntactic manifestation (by Smadja and McKeown), or their semantic manifestation (by Nirenburg et al.). We find that these representations help in applying collocational constraints during generation, but without any further generalizations on use or meaning, they are not very useful for coming up with different ways of combining words. The representations just describe collocational behaviour.

The more recent and substantial lexical formalisms that we looked at sought to find generalizations in behaviour so that collocations can be used productively in generation systems. Each type of collocation, thus discovered, is best represented in a different way. So, for free-combining words, normal lexical semantics suffices. For the semantic collocations, investigated by Viegas and Bouillon, a mechanism that models explicitly the semantic regularity in word combination is necessary. The qualia structure attributed to a word in the Generative Lexicon specifies how it may combine with other words to express a desired meaning dependent on the context. The words are still combined according to the regular rules of grammar though. The more idiosyncratic lexical collocations seem to require less explicitness when it comes to semantics, but more for syntax, so they are modelled adequately by lexical functions. A lexical function represents an abstract semantic meaning, but plays a specific syntactic role. All the lexical functions for a word must be listed exhaustively, but they can be used productively because of their systematic semantic organization. Finally, we find that for some collocations, their abstract meaning is so specific that they should not, or cannot, be generalized. These are best represented simply as idioms.

Just as phrasal patterns are intimately connected to syntactic structure, collocations are as well. We see, especially in the MTM and Wanner's system, but also in Spokesman, that text plans can be built using abstractions of lexical knowledge (in the form of lexical functions). This means that the text planner can be aware of the lexical means of expressing a concept, or set of concepts, while not actually constraining itself to choose a specific structure because of a specific word choice.

In Spokesman, complex lexical choice is not dealt with, but in the Generative Lexicon, the MTM, and Wanner's system, lexical choice is at least an implicit issue. The three formalisms strive to give as complete a set as possible of all the options available to express a semantic input. The problem, as we have seen, is that although they do give many options, they do not give any distinctions between them. So, in these systems, actual lexical choice was reduced to a simplistic process, or entirely ignored.

It was not possible to design anything more sophisticated because the distinctions between options provided by the lexicon were largely a matter of factors outside of semantics. For instance, two options might differ rhetorically (here, Wanner's system generates text with respect to a perspective such as action-oriented) or stylistically, might carry different connotations or implicatures, etc.

In section 2.3 we saw that research into modelling this kind of knowledge has only just begun, and is yet to be well understood. One thing is clear though: an expressive generator will need command of pragmatic knowledge. Thus, a lexical choice process, and even a process to assemble words into phrases (as the above systems do), has to use more than semantic factors. If factors such as denotation, connotation, emphasis, implicature, and formality (from Hirst) are to be implemented, the implementation must be done with an eye towards usage. It is not enough to simply describe words with, say, stylistic features, because this leads to ad hoc mapping algorithms, like the one in Nirenburg's DIONYSUS. Hirst's proposal that differences between words, concepts, and (I would add) word combinations, be represented as objects that can be reasoned about is good, if knowledge about usage can be incorporated into the entries of the words.

One can then envision a lexical choice process that works in tandem with knowledge about word usage and combination. Thus it would need to consider not only semantic constraints, but constraints and preferences pertaining to structural and pragmatic factors also. We have the ingredients for such a process, even though some may not be well understood; what remains is the very difficult task of integration.

We can take this idea even further in considering the whole of text generation. I advance the call, once again, for a *phrasal lexicon*. Instead of containing simple phrasal-patterns as the implementations described in section 2.2.3 did, a phrasal lexicon should contain patterns ranging from words all the way up to rhetorical structures, just as Wanner's model does. Perhaps such a lexicon would be better called a *structural lexicon* (compare to Fillmore's Construction Grammar (Fillmore, Kay and O'Connor, 1988)). Just as collocations can be organized in a hierarchy of idiosyncrasy, so can phrases, syntactic structures, and rhetorical structures. Items in the lexicon should also be given pragmatic and stylistic knowledge as discussed above. Of course, the knowledge should not be solely descriptive. Explicit functional knowledge about the choice, usage, and formation of each type of structural pattern on each of its levels of idiosyncrasy (just as for collocations) is necessary.

It may be a tall order, but if lexical knowledge is taken to be intimately connected to the expression of concepts in a generator's input, and to the structure, both syntactic and rhetorical, of its output, then text generation needs to be an intensive knowledge-driven process. A phrasal lexicon would seem to be a good means of representing the required knowledge.

References

Anick, Peter and Bergler, Sabine (1992). Lexical structures for linguistic inference. In Pustejovsky, James and Bergler, Sabine, editors, *Lexical Semantics and Knowledge Representation: First SIGLEX Workshop*, Lecture Notes in Artificial Intelligence 627, pages 121–135. Springer-Verlag.

Becker, Joseph D. (1975). The phrasal lexicon. In Schank, R. and Nash-Webber, B.L., editors, *Theoretical Issues in Natural Language Processing*, pages 70–73, Cambridge, Mass.

Benson, Morton (1990). Collocations and general-purpose dictionaries. *International Journal of Lexicography*, 3(1):23–35.

Benson, Morton; Benson, Evelyn; and Ilson, Robert (1986). *The BBI Combinatory Dictionary of English: A Guide to Word Combinations*. J. Benjamins Pub. Co., Amsterdam.

Cruse, D. Alan (1986). *Lexical Semantics*. Cambridge University Press.

Danlos, Laurence (1987). *The Linguistic Basis of Text Generation*. Cambridge University Press.

DiMarco, Chrysanne; Hirst, Graeme; and Stede, Manfred (1993). The semantic and and stylistic differentiation of synonyms and near-synonyms. In *AAAI Spring Symposium on Building Lexicons for Machine Translation*, pages 114–121, Stanford, CA.

Fillmore, Charles J.; Kay, Paul; and O'Connor, Mary Catherine (1988). Regularity and idiomaticity in grammatical constructions: The case of 'let alone'. *Language*, 64(3):501–538.

Goldman, N. M. (1975). Conceptual generation. In Schank, Roger, editor, *Conceptual Information Processing*, pages 289–371. North-Holland, Amsterdam.

Grosz, Barbara J.; Sparck Jones, Karen; and Webber, Bonnie Lynn, editors (1986). *Readings in Natural Language Processing*. Morgan Kaufmann Publishers.

Halliday, Michael A. K. (1985). *An Introduction to Functional Grammar*. E. Arnold, London.

Hirst, Graeme (1995). Near-synonymy and the structure of lexical knowledge. In *AAAI Symposium on Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity, and Generativity*, Stanford, CA. (to appear).

Iordanskaja, Lidija; Kittredge, Richard; and Polguère, Alain (1991). Lexical selection and paraphrase in a meaning-text generation model. In Paris, Cécile L.; Swartout, William R.; and Mann, William C., editors, *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, pages 293–312. Kluwer Academic, Boston.

Jacobs, Paul S. (1985). PHRED: A generator for natural language interfaces. *Computational Linguistics*, 11(4):219–242.

Knight, Kevin and Luk, Steve K. (1994). Building a large-scale knowledge base for machine translation. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, pages 773–779.

Kukich, Karen (1988). Fluency in natural language reports. In McDonald, David D. and Bolc, Leonard, editors, *Natural Language Generation Systems*, pages 280–311. Springer-Verlag.

Mann, William C. and Thompson, Sandra A. (1987). Rhetorical structure theory: Description and reconstruction of text structures. In Kempen, Gerard, editor, *Natural Language Generation*, pages 85–95. Martinus Nijhoff Publishers, Dordrecht.

Matthiesen, Christian (1991). Lexico(grammatical) choice in text generation. In Paris, Cécile L.; Swartout, William R.; and Mann, William C., editors, *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, pages 249–292. Kluwer Academic, Boston.

McDonald, David and Busa, Federica (1994). On the creative use of language: The form of lexical resources. In *Proceedings of the Seventh International Workshop on Natural Language Generation*, pages 81–90, Kennebunkport, Maine.

McDonald, David D. (1983). Description directed control: Its implications for natural language generation. In Cercone, Nick, editor, *Computational Linguistics*, International Series in Modern Applied Mathematics and Computer Science 5, pages 111–129. Plenum Press, New York. Reprinted in (Grosz, Sparck Jones and Webber, 1986).

McDonald, David D. (1987). Factors contributing to efficiency in natural language generation. In Kempen, Gerard, editor, *Natural Language Generation*, pages 159–181. Martinus Nijhoff Publishers, Dordrecht.

McKeown, Kathleen R. (1985). *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge University Press.

Mel'čuk, Igor (1981). Meaning-text models: A recent trend in Soviet linguistics. *Annual Review of Anthropology*, 10:27–62.

Mel'čuk, Igor; Arbatchew-Jumarie, N.; Elnitsky, L.; Iordanskaja, L.; and Lessard, A., editors (1984). *Dictionaire Explicatif et Combinatoire du Français Contemporain.* Les Presses de l'Université de Montréal, Montréal.

Mel'čuk, Igor and Polguère, Alain (1987). A formal lexicon in meaning-text theory (or how to do lexica with words). *Computational Linguistics*, 13(3–4):261–275.

Mel'čuk, Igor and Zholkovsky, Alexander (1988). The explanatory combinatorial dictionary. In Evens, Martha, editor, *Relational Models of the Lexicon: Representing Knowledge in Semantic Networks*, pages 41–74. Cambridge University Press.

Meteer, Marie (1992). *Expressibility and the Problems of Efficient Text Planning.* Pinter Publishers, London.

Nirenburg, Sergei; Carbonell, Jamie; Tomita, Masaru; and Goodman, Kenneth (1992). *Machine Translation: A Knowledge-Based Approach.* Morgan Kaufmann Publishers, Inc.

Nirenburg, Sergei and Defrise, Christine (1992). Application-oriented computational semantics. In Rosner, Michael and Johnson, Roderick, editors, *Computational Linguistics and Formal Semantics*, pages 223–256. Cambridge University Press.

Nirenburg, Sergei; Lesser, Victor; and Nyberg, Eric (1989). Controlling a language generation planner. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, pages 1524–1530.

Nirenburg, Sergei and Nirenburg, Irene (1988). A framework for lexical selection in NLG. In *Proceedings of the 12th International Conference on Computational Linguistics (COLING-88)*, pages 471–475.

Onyshkevych, Boyan A. and Nirenburg, Sergei (1992). Lexicon, ontology, and text meaning. In Pustejovsky, James and Bergler, Sabine, editors, *Lexical Semantics and Knowledge Representation: First SIGLEX Workshop*, Lecture Notes in Artificial Intelligence 627, pages 289–303. Springer-Verlag.

Penman Natural Language Group (1988). The Nigel manual. Technical report, Information Sciences Institute of the University of Southern California.

Penman Natural Language Group (1989). The Penman reference manual. Technical report, Information Sciences Institute of the University of Southern California.

Pustejovsky, James (1991). The generative lexicon. *Computational Linguistics*, 17(4):409–441.

Pustejovsky, James and Anick, Peter G. (1988). On the semantic interpretation of nominals. In *Proceedings of the 12th International Conference on Computational Linguistics (COLING-88)*, pages 518–523.

Pustejovsky, James; Bergler, Sabine; and Anick, Peter (1993). Lexical semantic techniques for corpus analysis. *Computational Linguistics*, 19(2):331–358.

Pustejovsky, James and Bouillon, Pierrette (1994). On the proper role of coercion in semantic typing. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, pages 706–711.

Pustejovsky, James and Nirenburg, Sergei (1987). Lexical selection in the process of language generation. In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, pages 201–206.

Ramsay, Allan (1994). The co-operative lexicon. In *Proceedings of the International Workshop on Computational Semantics*, pages 171–179, Tilburg.

Rösner, Dietmar and Stede, Manfred (1994). Generating multilingual documents from a knowledge base: The TECHDOC Project. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, pages 339–343, Kyoto, Japan.

Sanfilippo, Antonio; Benkerimi, Kerima; and Dwehus, Dagmar (1994). Virtual polysemy. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, pages 696–700.

Smadja, Frank and McKeown, Kathleen (1991). Using collocations for language generation. *Computational Intelligence*, 7:229–239.

Stede, Manfred (1993). Lexical choice criteria in language generation. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, pages 454–459, Utrecht.

Stede, Manfred (forthcoming). Lexicalization in natural language generation: A survey. *Artificial Intelligence Review*, to appear.

Teich, Elka (1992). Komet: Grammar documentation. Technical report, GMD/IPSI, Darmstadt, Germany.

Viegas, Evelyne and Bouillon, Pierette (1994). Semantic lexicons: The cornerstone for lexical choice in natural language generation. In *Proceedings of the Seventh International Workshop on Natural Language Generation*, pages 91–98, Kennebunkport, Maine.

Wanner, Leo (1994a). Building another bridge over the generation gap. In *Proceedings of the Seventh International Workshop on Natural Language Generation*, pages 137–144, Kennebunkport, Maine.

Wanner, Leo (1994b). On lexically biased discourse structure organization in text generation. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, pages 369–375.