Semantic Representations of Near-Synonyms for Automatic Lexical Choice

by

Philip Edmonds

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto

Abstract

Semantic Representations of Near-Synonyms
for Automatic Lexical Choice

Philip Edmonds
Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto
1999

We develop a new computational model for representing the fine-grained meanings of near-synonyms and the differences between them. We also develop a sophisticated lexical-choice process that can decide which of several near-synonyms is most appropriate in any particular context. This research has direct applications in machine translation and text generation, and also in intelligent electronic dictionaries and automated style-checking and document editing.

We first identify the problems of representing near-synonyms in a computational lexicon and show that no previous model adequately accounts for near-synonymy. We then propose a preliminary theory to account for near-synonymy in which the meaning of a word arises out of a context-dependent combination of a context-independent core meaning and a set of explicit differences to its near-synonyms. That is, near-synonyms cluster together.

After considering a statistical model and its weaknesses, we develop a clustered model of lexical knowledge, based on the conventional ontological model. The model cuts off the ontology at a coarse grain, thus avoiding an awkward proliferation of language-dependent concepts in the ontology, and groups near-synonyms into subconceptual clusters that are linked to the ontology. A cluster acts as a *formal usage note* that differentiates near-synonyms in terms of fine-grained aspects of denotation, implication, expressed attitude, and style. The model is general enough to account for other types of variation, for instance, in collocational behaviour.

We formalize various criteria for lexical choice as *preferences* to express certain concepts with varying indirectness, to express attitudes, and to establish certain styles. The lexical-choice process chooses the near-synonym that best satisfies the most preferences. The process uses an approximate-matching algorithm that determines how well the set of lexical distinctions of each near-synonym in a cluster matches a set of input preferences.

We implemented the lexical-choice process in a prototype sentence-planning system. We evaluate the system to show that it can make the appropriate word choices when given a set of preferences.

Acknowledgements

I was very fortunate to have Graeme Hirst as my PhD supervisor. His great insight meant that almost every one of his observations about the research presented in this thesis uncovered an important issue that required elaboration or clarification, or that even opened up whole new avenues to explore. Moreover, he was a main source of inspiration for this work.

I also owe thanks to Robert Dale. As external examiner, he was extremely thorough and constructive in his comments. The final version of the thesis owes much to his careful reading of the penultimate draft. Thanks also go to Jack Chambers, my second external examiner.

As my own thoughts evolved throughout the course of this work, so my advisory committee did. Thanks go to all of the members of my committee: in the final incarnation, Mark Chignell, John Mylopoulos, and Brian Merrilees, and in previous incarnations, Chrysanne DiMarco, Marilyn Mantei, Ray Reiter, and Jeff Siskind. All gave excellent advice.

Thanks to Manfred Stede, both for providing his MOOSE system, and for reading previous versions of the thesis.

For other help and inspiration along the way, I thank Melanie Baljko, Alex Budanitsky, Steve Green, Eduard Hovy, Daniel Marcu, and Leo Wanner.

Financial support came from the Natural Sciences and Engineering Research Council of Canada and the Government of Ontario (through their Ontario Graduate Scholarship program).

Finally, Lynne Gardham, who came with me on this rather long journey, neither of us comprehending the full scope of it, does and always will receive my utmost gratitude, admiration, and love—not the least of which is for making me get on with it.

Contents

List of Tables

List of Figures

List of Program Listings

Chapter 1

Introduction

A word is a complex entity that can, depending on the situation, express a myriad of implications, connotations, and attitudes on top of its basic 'dictionary' meaning. And a word often has many near-synonyms that differ only in these nuances of meaning. So, whether one is writing or speaking, in order to find the right word—the one that precisely conveys one's desired meaning, but avoids unwanted implications—one must carefully evaluate the differences between all of the options. This task is of course difficult for people, let alone computer systems.

Consider machine translation (MT), the quintessential application of computational linguistics. How would an MT system determine the best English word for the French *erreur* when there are so many possible slightly different translations? The system could choose *error, mistake, blunder, slip, lapse, boner, faux pas, boo-boo,* and so on, but the most appropriate choice is a function of how *erreur* is used (in context) and of the difference in meaning between *erreur* and each of the possibilities. So, not only would the system have to recover the nuances that *erreur* conveys in the particular context in which it has been used, but it would also have to find the word (or words) in the target language that most closely convey the same nuances in the context of the other words that it is choosing concurrently. An exact translation is probably impossible, for *erreur* is in all likelihood to be as different from each of its possible translations as they are from each other. That is, in general, every translation possibility will omit some nuance or express some other possibly unwanted nuance. Thus, faithful translation requires a sophisticated lexical-choice process that can determine which of several near-synonyms provided by one language for a word in another language is most appropriate in any particular situation.

Or, consider natural language generation (NLG), an application quickly gaining demand, whose purpose is to produce natural language text from a non-linguistic input. NLG can be considered as part of the process of an MT system that first converts text into an intermediate non-linguistic representation. So, a truly articulate NLG system, which we don't yet have, would also require a sophisticated lexical-choice process for the same reasons as MT systems do. The system would have to be able to reason about the potential effects of every available option.

Or, consider a new type of thesaurus for a standard word processor that, instead of presenting the user with a mere list of similar words, facilitates choice by ranking the possibilities according to their appropriateness in context and in meeting a few preferences specified by the user. Such an *intelligent thesaurus* would greatly benefit many writers, and would be a definite improvement over the simplistic thesauri in current word processors.

In all of these applications a comprehensive computational model of fine-grained lexical knowledge is necessary. The research presented in this thesis is about developing such a model, and about developing a computational lexical-choice process that uses the model.

1.1 Lexical choice

Lexical choice is the process of determining which word in a given language most precisely expresses a meaning specified in some formalism other than the given language itself. Its goal is to verbalize the exact denotation and connotation desired, and nothing else. The formalism might represent meaning as words in another natural language (e.g., in the lexical transfer approach to MT) or as (instances of) non-linguistic concepts defined in a model of world knowledge. The latter, more common approach (at least outside of commercial MT systems), is assumed in this thesis because of its generality over different kinds of natural-language applications. In this way, lexical choice is viewed as mapping from concepts to words, and thus relies crucially on the representation and organization of lexical knowledge, the central concern of *lexical semantics*.

Lexical choice is difficult for several reasons. First, the desired semantics can usually be ‘covered’ in different ways; that is, different sets of words can express the very same meaning. For example, *go by plane* is equivalent in semantics to *fly*, and *give* expresses the same situation as *receive* does but from a different perspective. Second, lexical choice and syntactic choice are inherently interdependent since the syntactic context constrains word choice, which in turn constrains syntactic choice. For example, Meteer (1992) gives the example that one can say *make an important decision* but not *decide importantly*: the verb *decide* constrains the syntactic structure so that importance cannot be expressed, but a different lexical choice, *make a decision*, does not have this constraint. Third, lexical constraints—constraints arising from words rather than syntax—can influence choice. These range from (semantic) selectional restrictions to seemingly idiosyncratic collocational constraints. For example, the verb *blab* ‘reveal a secret’ requires its agent to be human, whereas *reveal* can take an inanimate agent. And the adjective *daunting* collocates with *task* but not with *job* (see example 1.5 below).

Synonymy as a problem in lexical choice has not yet been seriously considered in MT systems because other significant problems, like those above, had to be overcome first. The main goal has been to develop systems that could convey or translate the general meaning without concern for fine nuances of semantics or style. In fact, the early systems, such as MUMBLE (McDonald, 1983) and TEXT (McKeown, 1985), hardly ‘chose’ words at all because concepts and words were related one-to-one. But recent work on lexical choice in NLG systems has focused on representing the lexical knowledge necessary to enable the production of a wide variety of complex paraphrases, both structural and lexical, of the same input. For example, in both ADVISOR II (Elhadad, McKeown and Robin, 1997) and MOOSE (Stede, 1999), a sentence-planning stage has been proposed to deal with all of the interdependent decisions needed in constructing a sentence, including lexical choice. Structural and lexical variation is made possible by promoting the lexicon to a central role as a bridge between language-independent (or language-neutral) concepts and language-dependent structures. However, even though these systems can represent some forms of variation, they lack the complementary ability to actually choose *which* paraphrase or word is most appropriate. With few exceptions, near-synonyms are treated as absolute synonyms, and so are simply listed as equivalent options. But now that we have the infrastructure, work on sophisticated lexical choice is poised to proceed.

By introducing synonymy to the lexicon, lexical choice is no longer simply the problem of mapping from concepts to words: it is recast into choosing from among alternatives the right word; that is, *genuine* lexical choice.

Error implies a straying from a proper course and suggests guilt as may lie in failure to take proper advantage of a guide . . .

Mistake implies misconception, misunderstanding, a wrong but not always blameworthy judgment, or inadvertence; it expresses less severe criticism than *error*.

Blunder is harsher than *mistake* or *error*; it commonly implies ignorance or stupidity, sometimes blameworthiness.

Slip carries a stronger implication of inadvertence or accident than *mistake*, and often, in addition, connotes triviality.

Lapse, though sometimes used interchangeably with *slip*, stresses forgetfulness, weakness, or inattention more than accident; thus, one says a *lapse* of memory or a *slip* of the pen, but not vice versa.

Faux pas is most frequently applied to a mistake in etiquette.

Bull, howler, and boner are rather informal terms applicable to blunders that typically have an amusing aspect.

Figure 1.1: An abridged entry from *Webster's New Dictionary of Synonyms* (Gove, 1973).

1.2 Near-synonymy

Near-synonymy—the property of several words that are very close in meaning, yet not identical, for whatever reason (see chapter 2)—adds many complications to lexical choice. Near-synonyms by their nature lead one to compare words and so highlight the *differences* in meaning rather than just positive features. Many dictionaries of synonym discrimination have been published that do just this, including *Webster's New Dictionary of Synonyms* (Gove, 1973) and Hayakawa's *Choose the Right Word* (Hayakawa, 1994). Figure 1.1 shows a typical entry, essentially a usage note, for the near-synonyms of *error*. (Throughout this thesis I will rely on usage notes from these reference books and others, and not merely on my own intuitions.)

One of the main problems for lexical choice with regard to synonymy is that while the differences between near-synonyms can be very subtle, the overall effect of using one near-synonym instead of another can be significant. Consider the options in each of 1.1 through 1.5 below, which illustrate several types of lexical difference. (The samples were taken from text corpora; the (a) selection of the first three examples is taken from the *Brown Corpus*, and of the last two from the *Wall Street Journal Corpus*. The (b) and (c) selections, where included, give possible substitutions.)

1.1a. Hudson's first *error* of the fourth voyage occurred only a few miles down the Thames.

b. Hudson's first *blunder* of the fourth voyage occurred only a few miles down the Thames.

1.2a. "You'll stay right here", *commanded* Bill Doolin, covering Red with his rifle.

b. "You'll stay right here", *ordered* Bill Doolin, covering Red with his rifle.

1.3a. Was the man *drunk* or crazy or both?

b. Was the man *inebriated* or crazy or both?

c. Was the man *pissed* or crazy or both?

1.4a. The current campaign is just the first leg of an *aggressive* three-to-five-year direct marketing plan.

b. The current campaign is just the first leg of a *pushy* three-to-five-year direct marketing plan.

1.5a. Mr. Hamilton faces a daunting *task* in reviving Ogilvy's stagnant New York office.

b. Mr. Hamilton faces a daunting *job* in reviving Ogilvy's stagnant New York office.

In 1.1 the major difference is semantic, as figure 1.1 shows. Choosing *blunder* would indeed have been a blunder had the writer not intended to call Hudson stupid. In 1.2 the difference is also semantic, but more subtle. The verb *order* tends to imply a peremptory (or arbitrary) exercise of authority, and *command* an official exercise (Gove, 1973). The overall effect in this case is subtle, perhaps too subtle to make issue over; a different relationship may be implied between the two participants, though it need not always be the case.

The difference in 1.3 is due mostly to the style or tone of the three terms. In this case, the semantics of each are the same, while the formality is different (though *inebriated* can imply an intoxication that results in exhilaration (Gove, 1973)). Depending on the overall context, the different levels of formality will have different pragmatic effects. For instance, choosing the informal *pissed* in a formal context could be used for irreverence or special emphasis.

In 1.4b, *pushy* expresses a derogatory attitude of the speaker towards the "marketing plan" that *aggressive* does not in 1.4a (Gove, 1973). An advertiser would almost never want a *pushy* campaign. Notice that, unlike a stylistic distinction, which pertains strictly to the word itself, an attitude is always a relation between participants of the situation. So, for instance, in 1.4, the attitude involves the speaker and the "marketing plan".

Finally, in 1.5 the difference has to do not with meaning (in its normal sense of denotation), but rather with the collocational patterns of *task* and *job*. The expressions *face a task* and *daunting task* are more common than *face a job* and *daunting job*. It is hard to determine if collocations have an underlying semantic cause, as selectional restrictions do, but *face a daunting job* does sound odd.

Of course, near-synonyms can often be intersubstituted with no apparent change of effect on an utterance, but it is very difficult to specify which contexts will cause an effect and which won't. Lexicographers, for instance, resort to using 'frequency' terms such as *sometimes* and *usually* (see figure 1.1), and also provide examples of typical usage. Unfortunately, the interactions of context and lexical choice (not to mention other decision tasks in text production) are not very well understood as yet, and the few theories that do exist are not adequate for computational purposes (see chapter 4). But while we can't yet solve the problems of context-dependence, there are several other problems associated with representing and using near-synonyms that we can solve before we have a fully worked out theory of context. Of course, any solutions that we develop will have to recognize the eventual need to account for context.[1]

Now, as for the differences themselves, there are clearly several types including at least the following:

[1] One reason that we focused on lexical choice rather than lexical analysis is that we didn't need to have a complete theory of context. Lexical analysis, on the other hand, appears to require beforehand a more formalized account of the influence of context on the expression on nuance. Edmonds (1998) discusses the problem in more detail.

- fine-grained and fuzzy denotational differences (see examples 1.1 and 1.2)
- stylistic differences (see example 1.3)
- differences in expressed attitude of the speaker or writer (see example 1.4)
- differences in collocations with other words (see example 1.5)

Each type calls for its own manner of representation in the lexicon, and, furthermore, the input to the lexical choice process will have to account for additional kinds of criteria (e.g., what style to use, or what attitude to express) on top of the usual conceptual input.

Near-synonyms are often said to differ in terms of *connotations*—loosely, ideas that colour the meaning of a word—but it is difficult to classify connotational differences as a separate category, because they seem to cut across the various other types—the term is ambiguous. Sometimes it is used to refer to any non-denotational component of meaning (including style and affect), but often a semantic distinction is said to be connoted, e.g., *slip* connotes triviality (see figure 1.1). The one aspect that distinguishes connotation, though, is that it refers to meaning conveyed *indirectly* by mere suggestion or implication. Such indirect meanings are usually peripheral to the main meaning conveyed by an expression, and it is usually difficult to ascertain definitively whether or not they were even intended to be conveyed.

Actually, indirect meanings make up a large portion of the nuances conveyed by words, as an analysis of the reference books reveals (see chapter 3). For instance, the verb *order* only suggests peremptory authority; it is not a necessary implication of the word (Gove, 1973). Or, the difference between *correct* and *right* is that the latter adds just a hint of moral approval (Hayakawa, 1994). In both instances, the listener has the license to decide that the nuance was not intended at all. Contrast this with the classical view of Frege (1892) or Tarski (1944) that a word denotes a concept (or extension of a concept) represented by a set of features that are individually necessary and collectively sufficient to define the concept. Most computational systems presume this view because it simplifies the concept–word link. In generation systems, for instance, the features amount to the necessary applicability conditions of a word. Clearly, indirect meanings are not necessary features of a word, but the classical view is inadequate for representing them (see chapter 4).

Now, each type of distinction between two or more near-synonyms can be considered a separate *dimension*. (Note that I am using ‘distinction’ to refer to a single differentia between two words, and ‘difference’ to refer, more loosely, to the overall difference between the words, or to a complete set of distinctions.) For instance, the ‘error’ words can be placed on a continuous dimension of severity of criticism, and on a binary (i.e., two-value) dimension of misconception. The problem is that near-synonyms can vary on several *dimensions* at the same time (DiMarco, Hirst and Stede, 1993). Indeed, multi-dimensional differences seem to be the norm—most of the above examples differ on at least two dimensions—and the analysis of the reference books confirms this (see chapter 3).

There are several reasons why multi-dimensional differences present a problem for lexical choice. First, there is a serious problem with lexical ‘gaps’. Given a set of criteria for choice, say, as values on several dimensions, there might not be a word that matches exactly. That is, the multi-dimensional space of words is sparsely populated. To solve this, the lexical choice process could ignore some criteria, could find the closest match, or it could produce a phrase instead. For example, there is no word in English for ‘error’ that expresses both misconception and very severe criticism; *mistake* expresses the former, and *blunder* the latter, but the phrase *serious mistake* might be a good compromise. A related problem is unwanted implications. If one ignores a criterion, or simply chooses a word because of its value on a particular dimension, then one might also imply possibly unwanted ideas on other dimensions. For example, it might not be possible to find a word for ‘error’ that expresses low severity of criticism and yet does not express misconception. Adding extra words, as with lexical gaps, can sometimes mitigate an unwanted meaning (e.g., *inadvertent error,* in this case).

Another problem is that a set of dimensions may be interdependent. For example, *forest* and *woods* shade into one another in terms of size, wildness, and proximity to civilization (Room, 1981). Not only are the denotations of these words inherently fuzzy, but the dimensions are correlated, and may even be dependent on one another. For example, it might not make sense to ask for a word for ‘tract of trees’ that is large, far from civilization, and yet not wild, because such a thing does not exist and hence has not been named. In this case, the relation between the dimensions has to do with world-knowledge about trees, forests, and so on, but the relationship can also be semantic. For instance, the misconception expressed by *mistake* may be related to its non-blameworthiness. In either case, the lexical choice process must have the ability to reason about the relationships between dimensions.

Admittedly, it is probably wrong to think of dimensions (or values on dimensions) as simple features attached to words, because a standard feature-based theory of meaning, such as Katz and Fodor’s (1963), would not account for the potentially complex relationships between dimensions. But there are more reasons for abandoning a purely feature-based account. For one, the inherent fuzziness of some denotations (e.g., of *forest* and *woods,* and of *marsh, swamp, fen,* and *morass,* which differ in terms of size, density, and vegetation (Hayakawa, 1994)) precludes simple feature-matching. For another, a dimension also has a kind of internal structure in that it relates specific roles and aspects of a situation. For example, *mistake* does not just have the feature of ‘misconception’; it actually implies a *concept* of misconception that has an agent (the same as the agent of the *mistake*) and that is the cause of the *mistake.* Therefore, a distinction of meaning has a dual nature as a difference in value (perhaps fuzzy) on a dimension, and as difference in full-fledged concepts.

In summary, the problems that we must overcome in order to adequately represent the meanings of near-synonyms are the following:

- The different types of distinction, namely, semantic, stylistic, attitudinal, and collocational must each be handled in its own way.

- Nuances of meaning can be conveyed indirectly, and with varying strength. There appears to be a continuum of indirectness from ‘suggestion’ to ‘implication’ to ‘denotation’. (I.e., nuances are not always necessary and sufficient truth-conditions.)

- Meanings, hence differences, can be fuzzy.

- Differences can be multi-dimensional, and the dimensions can be interdependent.

- Differences are not just between simple features, but involve concepts that relate roles and aspects of a situation.

- Context influences the relevance of lexical differences.

Chapter 3 will present more evidence of these problems and chapter 4 will show that the existing computational models are not suitable for representing or choosing from among near-synonyms, because none of the models incorporates solutions to enough of these problems.

These problems for lexical representation also suggest that choosing the right near-synonym is not a matter of satisfying constraints, but rather of attempting to meet or satisfy a set of *preferences*—the right word is a tradeoff between possibly conflicting desires. (Stede (1995) comes to the same conclusion.) However, including preferences in the input to lexical choice poses its own problems for lexical choice, because preferences could be assigned levels of importance, and they could be satisfied to different degrees and perhaps even collectively by a set of words. Furthermore, conflicting and redundant preferences would have to be handled.

1.3 Goals of this research

The two central goals of this research are to (1) develop a model for representing the meanings of near-synonyms and the differences between them and (2) develop a sophisticated computational lexical-choice process that can choose from a group of near-synonyms the one that best achieves the desired effects in the current context.

There are four subgoals to achieving these objectives, stated below with various assumptions, but one assumption must be stated up front. It is not a goal of this research to build a complete lexicon or ontology, or to decide on how a particular group of near-synonyms differ, or even to decide which words are near-synonyms. That is a job for lexicographers and other trained professionals, and, in any case, is an ongoing lifelong process—consider that current dictionaries and thesauri represent centuries of accumulated lexical knowledge. The goal here is to provide the fundamental framework that will enable the representation of near-synonyms in a computational system.

A classification of near-synonymic variation

Before developing a model that accounts for near-synonymy, it is necessary to analyze the ways in which near-synonyms can differ. But even before this, we must settle on a suitable definition of the term ‘near-synonymy’. So, we will examine first how ‘near-synonymy’ has been defined in the past, and then define the term for our research (see chapter 2). We will then study the form and content of the usage notes in the various synonym discrimination in order to characterize and classify the many types of near-synonymic variation (see chapter 3). This study, along with a survey of the past research on lexical semantics and lexical choice (see chapter 4), will show the unsuitability of the current models for representing near-synonyms, at least for computational purposes.

A computational model for representing fine-grained lexical knowledge

I will focus on near-synonymy in developing a model for representing fine-grained lexical knowledge that can be used in any computational application that requires a knowledge of fine-grained word meanings. Thus, the goal is to develop a practical model that supports automatic text analysis and automatic lexical choice rather than a general theory of lexical semantics. Psychological plausibility of the model is not of central concern either. The model will, however, require a new synthesis of research in lexical semantics, which will have implications for theories of lexical semantics in general (see chapter 5).

We will approach the problem in two ways. First, we will explore the use of a statistical model to represent differences in the usage of near-synonyms as differences in the co-occurrence of near-synonyms with other words in large text corpora (see chapter 6). Second, we will attempt to extend the conventional symbolic knowledge-based model of lexical knowledge, which represents lexical knowledge ultimately in terms of concepts in a structured database of conceptual and world knowledge, i.e., an *ontology* (see chapter 7). The new model will neither be specific to a particular domain of language use nor be dependent on a particular ontology. However, an implementation of the model will require a specific ontology and representational formalism.

A lexical-choice process

I will develop a model of lexical choice broadly applicable in MT systems, NLG systems, writer's aids, and authoring tools. In order to focus on lexical choice and near-synonymy, I will develop the process within the framework of an existing NLG system, MOOSE (Stede, 1999), that provides the essential infrastructure. The lexical-choice process will then be one of several interrelated processes in the sentence-planning stage of text production (see chapters 8 and 9). Of course, this in no way precludes the model or process being applied to other MT or NLG systems.

The input to the model will consist of an instantiation of a situation, which specifies semantic constraints for sentence planning, and a set of preferences to imply, emphasize, or otherwise convey information, both semantic and stylistic. The form of the input will be specified in the thesis. The output will be a structured sentence plan that can be realized by a separate existing surface generation component, such as Penman (Penman Natural Language Group, 1989).

Note that it is not the job of lexical choice (or the sentence planner) to decide why a certain idea should be implied or emphasized, or why a certain style should be used. These decisions are assumed to have already been made by either a sentence- or text- understanding process (in the case of an MT system), or by a text-planning process (in an NLG system), or by a human user. Thus, the process will not account directly for how overall 'pragmatic' effects can be achieved through lexical choice.

Implementation and evaluation

As proof of concept, I will implement a prototype system, based on MOOSE, that incorporates the knowledge-based model and the lexical-choice process. I will represent enough groups of near-synonyms that I can evaluate the system both to demonstrate the full range of its capabilities and to substantiate my claim that the model adequately accounts for near-synonymy (see chapter 10).

1.4 Overview of the contributions

The next two sections briefly outline the two major contributions of this thesis, and the final section gives an example of the capabilities of our implemented system, **I-Saurus**.

1.4.1 A clustered model of lexical knowledge

The first major contribution of this research is a new model of lexical knowledge representation called the *clustered model of lexical knowledge*, which is based on a theory that the meaning of a word arises out of a combination of an inherent context-independent denotation and a set of explicit differences to its near-synonyms (see chapter 5). Figure 1.2 depicts the clustered model. The top part of the diagram is an ontology of concepts related by inheritance. The lower part

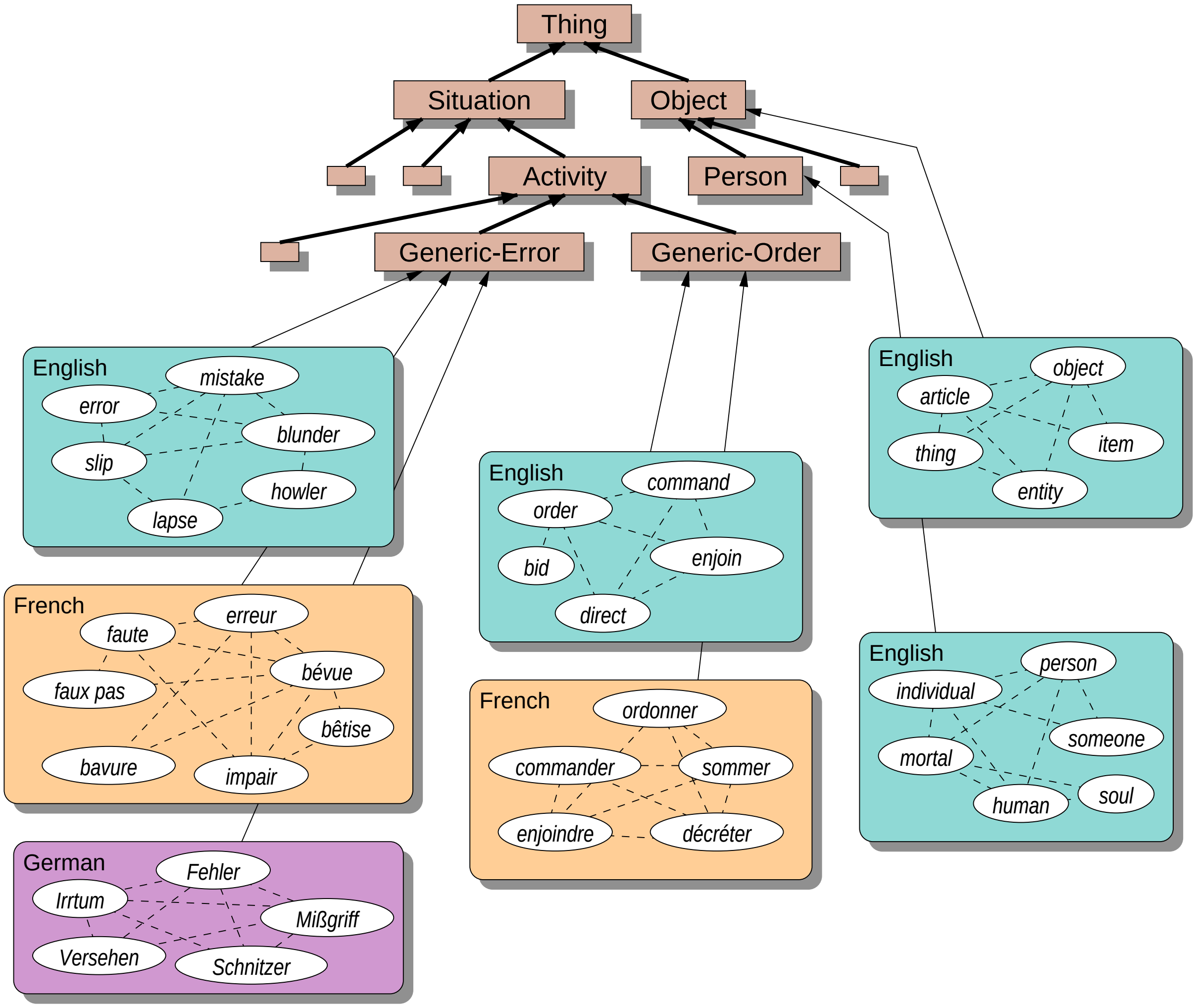


Figure 1.2: Depiction of the clustered model of lexical knowledge.

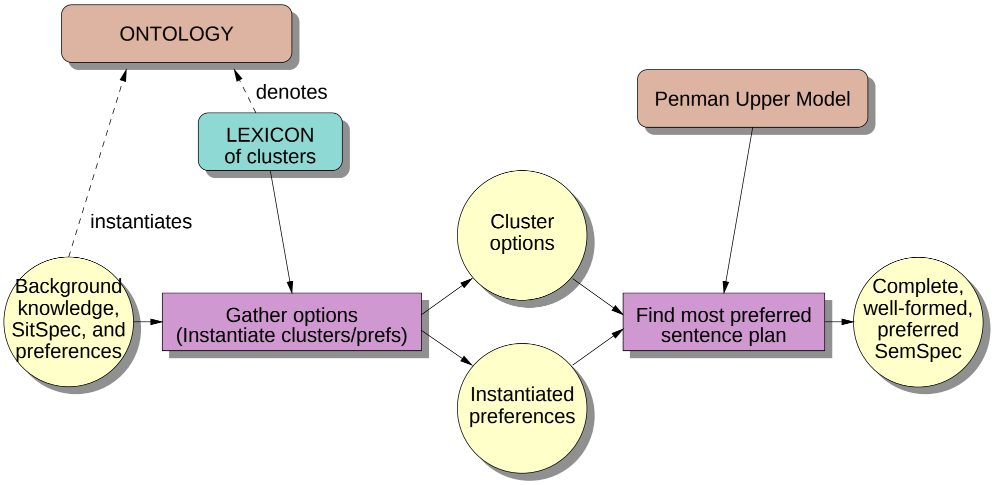

ONTOLOGY
denotes
LEXICON of clusters
Penman Upper Model
instantiates
Cluster options
Background knowledge, SitSpec, and preferences
Gather options (Instantiate clusters/prefs)
Find most preferred sentence plan
Complete, well-formed, preferred SemSpec
Instantiated preferences

Figure 1.3: Overall system architecture of I-Saurus.

shows several clusters of near-synonyms. A cluster represents lexical meaning on three levels; however, the figure depicts only the top and middle levels. At the top level, a cluster has a *core denotation* that represents the essential shared denotational meaning of its near-synonyms in terms of concepts in the ontology. At the middle level, the cluster represents semantic, stylistic, and expressive distinctions between the near-synonyms in terms of *peripheral concepts* (i.e., concepts, formed out of simple concepts in the ontology, that represent meanings that can be conveyed indirectly by words in the cluster), and various stylistic and attitudinal dimensions. The figure depicts these distinctions as mere undifferentiated dashed lines, though in the system they are represented by different kinds of structures. At the bottom level, not shown here, syntactic frames represent how each near-synonym can be combined with other words (compositionally) to form complete sentences. The model is described in chapter 7.

Listings 1.1 and 1.2 show the actual representations of two clusters in our system: the `error_C` and `err_C` clusters. (Lexical entries containing the syntactic frames are not shown.)

1.4.2 Lexical choice

The second major contribution is a lexical-choice process that uses the lexical knowledge that can be represented in the clustered model to rank possible word choices according to a set of preferences given as input. The process is embedded in a overall system called I-Saurus, whose architecture is depicted in figure 1.3. The input to the system consists of a specification of the situation to be verbalized, defined as a configuration of instances of concepts, and a ranked set of preferences to convey extra semantic information or attitudes, or to use a certain style (see table 1.1 below). The output is a sentence plan in sentence plan language (SPL) that can be realized by the Penman surface language generator (Penman Natural Language Group, 1989).

Lexical choice is performed in two main stages. In the first stage, each of the clusters of near-synonyms that can realize a portion of the input situation is instantiated as a cluster option. Then, with all of the options ready, the second stage attempts to choose a set of clusters options and a set of words (one from each chosen cluster option) that can be assembled into a well-formed sentence plan, and that collectively satisfy as many of the preferences as possi-

```
(defcluster error_C
    ;;; from Gove
    :syns (error_l mistake_l blunder_l slip_l lapse_l howler_l)
    :core (ROOT Generic-Error)
    :covers (ROOT)
    :p-link ((V1 (:and (Person V1) (ACTOR ROOT V1)))
             (V2 (:and (Deviation V2) (ATTRIBUTE ROOT V2))))

    :periph ((P1 Stupidity (ATTRIBUTE-OF V1))
             (P2 Blameworthiness (ATTRIBUTE-OF V1))
             (P3 Criticism (ACTEE V1) (ATTRIBUTE (P3-1 Severity)))
             (P4 Misconception (CAUSE-OF V2) (ACTOR V1))
             (P5 Accident (CAUSE-OF V2) (ACTOR V1))
             (P6 Inattention (CAUSE-OF V2) (ACTOR V1)))

    :distinctions (
     ;;  Blunder commonly implies stupidity.
     (blunder_l usually medium implication P1)

     ;;  Mistake does not always imply blameworthiness, blunder sometimes.
     (mistake_l sometimes medium implication (P2 (DEGREE ’medium)))
     (blunder_l sometimes medium implication (P2 (DEGREE ’high)))

     ;;  Mistake implies less severe criticism than error.
     ;;  Blunder is harsher than mistake or error.
     (mistake_l always medium implication (P3-1 (DEGREE ’low)))
     (error_l always medium implication (P3-1 (DEGREE ’medium)))
     (blunder_l always medium implication (P3-1 (DEGREE ’high)))

     ;;  Mistake implies misconception.
     (mistake_l always medium implication P4)

     ;;  Slip carries a stronger implication of accident than mistake.
     ;;  Lapse implies inattention more than accident.
     (slip_l always medium implication P5)
     (mistake_l always weak implication P5)
     (lapse_l always weak implication P5)
     (lapse_l always medium implication P6)

     ;;  Blunder expresses a pejorative attitude towards the person.
     (blunder_l always medium pejorative V1)

     ;;  Blunder is a concrete word, error and mistake are abstract.
     (blunder_l high concreteness)
     (error_l low concreteness)
     (mistake_l low concreteness)

     ;;  Howler is an informal term
     (howler_l low formality))
    )
```

Listing 1.1: The representation of the *error* cluster.

```
(defcluster err_C
    ;;; from my intuition (cf. Gove on error)
    :syns (err_l blunder-v_l)
    :core (ROOT Make
                (ACTOR (V1 Person))
                (ACTEE (V2 Generic-Error)))
    :covers (ROOT V2)
    :periph ((P1 Stupidity (ATTRIBUTE-OF V1)))

    :distinctions (
     (blunder-v_l usually medium implication P1)
     (blunder-v_l always medium pejorative V1)
     (err_l high formality))
    )
```

Listing 1.2: The representation of the *err* cluster.

ble. The lexical-choice process uses an approximate-matching function that attempts to find the closest match between a word's distinctions and the input preferences. The complete process is described in chapter 9.

1.4.3 An example of the system in use

This section presents a brief example that demonstrates some of the capabilities of I-Saurus; chapter 10 gives more detailed examples that demonstrate all of the system's capabilities. Consider the following input to the system, which includes both background knowledge about a situation and a specification of the parts of the situation that must be explicitly expressed in the output:

Background knowledge:

```
(John john1) (Activity error1)  (Deviation deviation1)
(ACTOR error1 john1)  (ATTRIBUTE error1 deviation1)
(Make make1) (ACTOR make1 john1) (ACTEE make1 error1)
```

Specification:

```
(make1 (ACTOR john1)
       (ACTEE error1))
```

In this situation, which could be realized as "John makes a mistake", `john1` is an instance of the concept `Person`; `error1` is an instance of an `Activity` that has the `ATTRIBUTE` `Deviation`, which is the definition of an 'error' in our ontology; and `make1` is an instance of `Make`, which is defined as a kind of activity roughly denoting 'cause to occur'. (Ontological modelling is not the focus of this thesis, so the fact that the verb *make* might be a support verb with no real semantics is not of concern here.) The system can paraphrase both lexically (e.g., by choosing *error* instead of *mistake*) and structurally (e.g., by choosing "John errs" instead of "John makes a mistake"). Table 1.1 shows the actual output of the system when given different combinations of preferences as input. The preferences are numbered and listed at the bottom of the table, and each specify an operator (e.g., "imply" or "favour") and any one of an instance of concept, an

Case	Input preferences	Output
i	None	John errs.
ii	1	John blunders.
iii	2	John makes a mistake.
iv	3	John makes a lapse.
v	4	John makes a slip.
vi	5	John makes a mistake.
vii	6	John makes an error.
viii	7	John makes a blunder.
ix	8	John makes a blunder.
x	9	John blunders.
xi	10	John errs.
xii	11	John makes a howler.
xiii	12	John errs.
xiv	13	John makes an error.
xv	14	John makes a blunder.
xvi	1, 2	John makes a mistake.
xvii	1, 5	John makes a blunder.
xviii	1, 5, 9, 11	John makes a blunder.
xix	2, 5, 9, 11	John makes a mistake.
xx	5, 7	—

Preferences:

```
 1  (imply (stupid1 (ATTRIBUTE-OF john1)))
 2  (imply (misconceive1 (CAUSE-OF deviation1) (ACTOR john1)))
 3  (imply (inattention1 (CAUSE-OF deviation1) (ACTOR john1)))
 4  (imply (accident1 (CAUSE-OF deviation1) (ACTOR john1)))
 5  (imply (criticize1 (ACTEE john1) (ATTRIBUTE (severe1 (DEGREE ’low)))))
 6  (imply (criticize1 (ACTEE john1) (ATTRIBUTE (severe1 (DEGREE ’0.5)))))
 7  (imply (criticize1 (ACTEE john1) (ATTRIBUTE (severe1 (DEGREE ’high)))))
 8  (imply (blameworthy1 (ATTRIBUTE-OF john1) (DEGREE ’high)))
 9  (disfavour john1)
10  (favour john1)
11  (low formality)
12  (high formality)
13  (low concreteness)
14  (high concreteness)
```

Table 1.1: Output of the system given different sets of preferences.

attitude, or a stylistic dimension. The ‘meaning’ of the preferences should be self-evident, so we’ll leave the details for later in the thesis.

Each labelled case in table 1.1 illustrates a facet of the choice algorithm. To understand why and how certain choices were made, refer back to listings 1.1 and 1.2 and to the following notes:

(i) When no preferences are given, the system makes the default choice, which, for this situation, is the verb *err*, since it is the first available option.

(ii)-(v) In these cases, each of the preferences is matched structurally against the representations of the near-synonyms in the cluster. For instance, in (ii), both the verb and the noun *blunder* have distinctions that match preference 1 (to imply that John is stupid), but the former is chosen by default. In (iii), the noun *mistake* is chosen because it has a distinction that matches preference 2 for implying a misconception.

(vi)-(ix) These cases illustrate the fuzzy matching capabilities of the system. The words *mistake*, *error*, and *blunder* differ on the dimension of severity of criticism of the actor of the error, but their positions on the dimension are fuzzy. That is, the meanings of words overlap to a certain extent. Thus, given the fuzzy set “low” in preference 5, the system finds that *mistake* matches best, but the other two words also match to a lesser degree (see case (xvii)).

(x)-(xi) These cases show that words can be chosen to express an attitude that the speaker has towards an entity of the situation. In (x), only *blunder* expresses the desired pejorative attitude towards John. In (xi), since none of the options expresses a favourable attitude, the system chooses *err* by default.

(xii)-(xv) Preferences to establish certain textual styles can also be specified. For example, *howler* is an informal word, so it satisfies preference 11 for low formality in (xii).

(xvi) The rest of the cases show the effects of specifying combinations of preferences. In (xvi), when preferences 1 and 2 are specified together, the system chooses *mistake* rather than *blunder*, because *mistake* has a better chance of satisfying preference 2 than *blunder* has of satisfying preference 1 (since *mistake* always implies misconception, and *blunder* only usually implies stupidity). There is no option that expresses both preferences.

(xvii) The opposite decision is made when preferences 1 and 5 are combined. Because *blunder* matches preference 5 a little bit (because of the fuzzy matching), whereas *mistake* does not match preference 1 at all, *blunder* edges out *mistake*.

(xviii) When we add to case (xvii) the preferences to both disfavour John (preference 9) and to use low formality (preference 11), the system must make a compromise. It doesn’t manage to satisfy the preference for low formality; only *howler* would do that, but *howler* would not at same time satisfy any of the other preferences. The system instead chooses *blunder*, because *blunder* satisfies the other preferences.

(xix) If we take case (xviii) and replace preference 1 with preference 2 (for implying a misconception), the system now sacrifices both preference 9 and preference 11. *Mistake* is the only word that satisfies even two of the four preferences; every other choice would satisfy only one preference.

(xx) This case inputs incompatible preferences to the system. It doesn't make sense to ask for both low and high severity at the same time. The system detects this incompatibility and outputs an error message (not shown in the table).

Chapter 2

Near-synonymy

Polysemy and synonymy are the two fundamental linguistic phenomena that influence the organization of the lexicon. But while polysemy has been given significant attention in recent research in lexical semantics and computational linguistics (Ruhl, 1989; Pustejovsky, 1995; Saint-Dizier and Viegas, 1995; Pustejovsky and Boguraev, 1996), synonymy has received relatively little, even though it is just as complex a phenomenon. This chapter discusses the problem of defining near-synonymy, and the following two chapters investigate the phenomenon of near-synonymy and review how it has been approached in research on lexical semantics and lexical choice.

Defining *synonymy* has proved to be very difficult. If one takes the view that synonyms are words that have the exact same meaning, it can be argued that synonyms do not and even cannot exist in any language. But if one relaxes this view to include words that are merely similar in meaning, then almost any two words can be considered synonyms at some level of granularity. Yet, under this view, synonymy can also be seen as so fundamental to a language that it influences the very structure of its vocabulary. This apparent paradox of synonymy—that synonymy is at the same time both impossible and fundamental in language—arises as a result of different approaches to defining synonymy. We begin by looking at absolute synonymy.

2.1 Synonymy as absolute synonymy

In their attempts to understand meaning and truth, philosophers have focused on synonymy as *absolute synonymy*, that is, intersubstitutability in all possible contexts without changing meaning, however they define 'meaning'. Quine (1951), for instance, considers several such definitions including interchangeability without change of truth value of the sentence (attributed to Leibniz) and interchangeability without affecting the set of logical entailments of the sentence (cf. Lyons's (1977) definition), but finds all such definitions problematic because of circularities resulting from the interrelated notions of meaning, synonymy, and truth. Quine argues that, in order to say that interchanging a word with its (putative) synonym in an expression does not change the meaning of the expression, one must have some way of specifying that the before and after expressions have the same meaning (or truth value, or whatever), and the only way to do this, according to Quine, is to reduce one expression to the other by interchanging synonyms. From a different perspective, Goodman (1952) instead argues that synonymy is impossible by using an equivalence of extension definition. He claims that for any terms *P* and *Q* one can always form the explicitly contrasting statement "a *P* which is not a *Q*". Goodman's conclusion relies on the idea that no two words can have the same extension, because one can always find a context, he would claim, in which two putative synonyms are not synonymous.

However, even if we assume that absolute synonymy is possible, pragmatic and empirical arguments show that it is very rare. Cruse (1986, p. 270) says that "natural languages abhor absolute synonyms just as nature abhors a vacuum", because the meanings of words are constantly changing. More formally, Clark (1992) employs her Principle of Contrast, that "every two forms contrast in meaning", to show that language works to eliminate absolute synonyms. Either an absolute synonym would fall into disuse or it would take on a new nuance of meaning.

2.2 Synonymy as a matter of degree

In any case, absolute synonymy is somewhat of a red herring for us, for if all of the options in a given context are completely identical there is obviously no need for non-random lexical choice. But, as we saw in chapter 1, in real situations of language production, there are usually several words to choose from that are *nearly* absolute synonyms, where a different choice would make a difference, however slight, in the overall expression. For us, these are the more interesting kinds of 'synonyms'. The question is then whether this relationship of *near-synonymy* is a phenomenon in its own right warranting its own special account. One could, of course, argue that if there are no (absolute) synonyms, then even very similar words should be treated like any other pair of words, similar or not. However, in the following sections, and continuing throughout the thesis, we will argue that near-synonymy is indeed a separately characterizable phenomenon of word meaning.

Near-synonyms are pervasive. In fact, every dictionary of synonyms actually contains only near-synonyms, a fact made obvious by those dictionaries that explicitly discriminate between synonyms. Lexicographers clearly take the view that synonymy is a matter of degree. But before we further investigate how lexicographers define near-synonymy, we will first listen to the semanticists.

Sparck Jones (1986) argues that only by defining synonymy as likeness of use in context can one come up with a workable definition. Hence, for her, if two or more *uses* of words are interchangeable in a particular context without changing the "ploy" (a primitive and intuitive notion "representing meaning, application, and form of a sentence"), those word-uses are synonymous. The same words need not be interchangeable in any other context. For example, we saw in section 1.2 that interchanging *command* and *order* in sentence 1.2 (page 3) had only a very subtle effect, if any, on the meaning of the sentence. But consider sentences 2.1a and 2.1b, in which interchanging the same two synonyms has a noticeable effect.[1]

2.1a. The doctor often *ordered* his patients to get plenty of rest.

b. The doctor often *commanded* his patients to get plenty of rest.

Sparck Jones argues that this type of synonymy is fundamental to the structure of the lexicon because it is required in order to give consistency and coherency to the infinite set of different uses of each word. Specifically, the set is structured into overlapping subsets of word-uses that have the same or nearly the same synonyms in a variety of contexts. (The more con-

[1] Synonymy is context-dependent in other ways too. For instance, some empirical research by Whitten, Suter and Frank (1979) has shown that human subjects judge differently the degree of synonymy of two words depending on the order in which the words are presented, which shows that an explicit context is not even necessary for context to be of influence.

texts in which the the uses of two words coincide, the higher the degree to which they are near-synonyms.) The structure thereby induced is that of a *thesaurus*.[2]

Now, one can question whether word-use is the right level at which to define synonymy, but the real problem here is Spark Jones's "ploy": how can we determine if the ploy of a sentence remains the same under intersubstitution of word-uses? The problem would seem the same as Quine's—indeed, Sparck Jones (1986, p. 86) concedes that "if we were to take overtones into account, we would never be able to make replacements", implying that even word-uses cannot be absolutely synonymous. Thus, Sparck Jones's definition of near-synonymy, since it relies on her definition of synonymy of word-use, is in the end unworkable.

Other semanticists have attempted to define synonymy and near-synonymy by focusing specifically on propositional meaning, since this is a prerequisite in formal semantics. Ullmann (1962), for instance, admits that there are few true synonyms, so he defines synonymy—really near-synonymy—imprecisely as similarity in "objective" meaning, but with possible differences on emotive, stylistic, or dialectal lines. For Lyons (1995), reiterating Lyons (1981), near-synonyms are terms that are "more or less similar, but not identical, in meaning", where meaning is taken to include both "descriptive" meaning (propositional meaning, or the set of logical entailments) and "expressive" meaning. While his definition is not very rigorous, Lyons carefully distinguishes this type of synonymy from *partial synonymy*, but it is not clear why. Partial synonyms fail to qualify as absolute synonyms in specific ways; either they are not "complete", that is, not identical "on all (relevant) dimensions of meaning" or they are not "total", that is not "synonymous in all contexts" (1981, p. 50). For instance, Lyons claims that *big* and *large* are partial synonyms because while being complete synonyms, they are not total synonyms; sentence 2.2b sounds odd, while 2.2a is fine. If we understand Lyons's dimensions of meaning as referring to only propositional dimensions, then many partial synonyms are words that differ in only propositional traits, whereas his near-synonyms can also differ in, or are perhaps required to differ in, expressive meaning. His definitions are not as lucid as they could be.

2.2a. You are making a *big* mistake.

b. You are making a *large* mistake.

Cruse (1986) also proposes two kinds of near-synonymy that are roughly equivalent to Lyons's types, only more rigorously defined. His *cognitive synonyms* are words that when intersubstituted in a sentence preserve its truth-conditions, but may change the expressive meaning, style, or register of the sentence, or may involve different idiosyncratic collocations. Thus, the two or more sentences obtained by substitution must logically entail one another, as shown in sentences 2.3a and 2.3b; *fiddle* and *violin* are thus cognitive synonyms. His *plesionyms*, on the other hand, change the truth-conditions, but still yield semantically similar sentences. For instance, since sentences 2.4a and 2.4b do not entail one another, *foggy* and *misty* are plesionyms.

2.3a. He plays the *fiddle* very well.

b. He plays the *violin* very well.

2.4a. It was *foggy* this morning.

[2] Many take a thesaurus to be a list of groups of synonyms, but Roget's original thesaurus (Chapman, 1992) was in fact a classification of words according to ideas—synonym groups are but a by-product. Sparck Jones claims that such a classification is the right one for machine translation, because it can aid in resolving lexical ambiguity; and I believe it is also the right overall structure for lexical choice, because it groups all of the options for lexical choice under the idea to be expressed.

Roget's International Thesaurus (Chapman, 1992)

> 354.11 Lie, falsehood, falsity, untruth, untruism, mendacity, prevarication, fib, taradiddle, flimflam, a crock; fiction, pious fiction, legal fiction; story, trumped-up story, farrago; yarn, tale, fairy tale, ghost story; farfetched story, tall tale, cock-and-bull story, fish story; exaggeration; half-truth, stretching of the truth, slight stretching, white lie, little white lie; a pack of lies. [Abridged.]

Webster's Collegiate Thesaurus (Kay, 1988)

> Lie: canard, cock-and-bull-story, falsehood, falsity, fib, inveracity, misrepresentation, misstatement, prevarication, story, tale, taradiddle, untruism, untruth.

Webster's New Dictionary of Synonyms (Gove, 1973)

> Lie, falsehood, untruth, fib, misrepresentation, story.

Figure 2.1: Comparison of entries for the synonyms of *lie* in three different dictionaries and thesauri.

b. It was *misty* this morning.

The need for two separate definitions is clear from the point of view of truth-conditional semantics because it at least allows synonymy—or one type of synonymy—to be defined, and it delineates the types of phenomena that the theory can account for. Unfortunately, it appears that the distinction is not so practical when it comes to representing the meanings of near-synonyms either in a dictionary or in a computational system. Variation in propositional meaning is only one of many types of variation, as we will see in chapter 3, but we would like to discriminate all types of near-synonym in the same entry. Having two definitions would not permit this. Moreover, a rigorous definition of cognitive synonymy is difficult to come up with, because it relies on the notion of granularity, which we will discuss below.

Lexicographers have always treated synonymy as near-synonymy. They define synonymy in terms of likeness of meaning, disagreeing only in how broad the definition ought to be. Traditionally, a synonym has been defined narrowly as one of two or more closely allied words differing only in minor ways (i.e., they must have the same, or very nearly the same, essential meaning), but a broader definition has included words that merely correspond in one or more characteristics of meaning (Egan, 1973, p. 23a).[3] Ultimately, the breadth of the definition depends on the purpose of the dictionary: a broad definition is better for word finding (such as *Roget's Thesaurus*), and a narrow definition for discrimination (such as *Webster's New Dictionary of Synonyms*). For example, consider the varying range of synonyms of *lie* taken in under the same entry in each of the thesauri shown in figure 2.1.

More explicitly, Roget followed the vague principle of "the grouping of words according to ideas" (Chapman, 1992, p. xiv). And since *Roget's Thesaurus* is structured hierarchically, words are ultimately grouped according to proximity of meaning: "the sequence of terms within a paragraph, far from being random, is determined by close, semantic relationships." (p. xiii).

[3] See Egan (1973) and Sparck Jones (1986), both of whom give excellent surveys of synonymy from the lexicographical perspective.

The lexicographers of *Webster's Collegiate Thesaurus* used the following more precise definition:

> A word is construed as a synonym if and only if it or one of its senses shares with another word or sense of a word one or more elementary meanings [. . . where an elementary meaning is a] discrete objective denotation uncolored by such peripheral aspects as connotations, implications, or quirks of idiomatic usage. (Kay, 1988, p. 9a)

And those of the *Webster's New Dictionary of Synonyms* used the following, similar, definition:

> A synonym, in this dictionary, will always mean one of two or more words in the English language which have the same or very nearly the same *essential* meaning. [. . .] the two or more words which are synonyms can be defined in the same terms up to a certain point. (Gove, 1973, p. 24a)

Clearly, the main point of these definitions is that near-synonyms must have the same essential meaning, but may differ in peripheral or subordinate ideas. Cruse (1986, p. 267) actually refines this idea and suggests that synonyms, of all types, are words that are identical in "central semantic traits" and differ, if at all, in only "peripheral traits".[4] But specifying formally just how much similarity of central traits and dissimilarity of peripheral traits is allowed can be a problem. That is, just what counts as a central trait and as a peripheral trait in defining a word? To discuss this further, we will now introduce the idea of granularity of representation.

2.3 Near-synonymy and granularity

The key to defining near-synonymy lies in the idea of *granularity of representation* of word meaning. By granularity we mean the level of detail used to describe or represent the meanings of words. A fine-grained representation is subtle, whereas a coarse-grained representation is crude. Consider this analogy: if one were to view a set of words (or, rather, the meanings of those words) from a distance, one would be able to see only their most basic, or coarse-grained, features, which, if they were the same, would lead one to think the words synonyms. But if one were to move closer to the words, the more subtle and fine-grained aspects of their meanings would emerge, thereby differentiating the words.

Granularity is independent of specificity, which is a property of concepts rather than representations of concepts. However, only the more general concepts would take part in a coarse-grained representation of another concept. A very general concept, say `Human`, can have, in a particular system, a very fine-grained representation, involving, say, a detailed description of the appearance of a human, references to related concepts such as `Eat` and `Procreate`, and information to distinguish the concept from other similar concepts such as `Animal`. Conversely, a very specific concept can have a very coarse-grained representation; one could represent the concept `Tree` at a level of detail only enough to say that it is a physical object.

Near-synonyms can occur at any level of specificity, but crucially it is the granularity of the representations of their meanings that enables one to distinguish one near-synonym from another. So, any definition of synonymy that does not take granularity into account is insufficient.

[4] For Cruse, a semantic trait of a word is a meaning of another word that participates in the meaning of the first word, but this is not important to the present discussion. Cruse also notes that one can rule out antonyms as being synonyms, because although they also share central traits, they do differ in one or two minor *contrastive* traits. True near-synonyms exhibit a *low* degree of contrastiveness.

Take Cruse's cognitive synonymy, discussed above. On the one hand, at a coarse grain of representation, any two words could be considered cognitive synonyms (because every word denotes a 'thing'). And on the other, no two words could ever be cognitive synonyms, because, even at a fine grain, apparent cognitive synonyms may be further distinguishable by a still more fine-grained representation. For instance, *drunk* and *inebriated* may seem to differ in only non-propositional traits, but *inebriated* can imply an intoxication that results in exhilaration (Gove, 1973), indicating that the words are in fact distinguishable at a finer grain. Thus, the pairs of words that are cognitive synonyms depends on the granularity with which we represent their propositional meanings.

By taking granularity into account, we can come up with a much more useful definition of near-synonymy, because we can now characterize the difference between essential and peripheral aspects of meaning. Assuming that we can set an appropriate level of granularity, the essential meaning of a word is the portion of its meaning that is representable only above that level of granularity, and peripheral meanings are representable only below that level.

But what is the appropriate level of granularity, the dividing line, as it were, between coarse-grained and fine-grained representations? We could simply use our intuition, or rather, the intuitions of lexicographers, which are filtered by some amount of objectivity and experience. It is practical to use the synonym lists contained in their dictionaries, because they already exist in relatively large amounts and because they were designed to help in word selection.

On the other hand, from a concern for the representation of lexical knowledge in a multilingual application, we can view words as (obviously, language-specific) specializations of language-independent concepts. For example, DiMarco, Hirst and Stede (1993) and Hirst (1995) presume a hierarchical organization of coarse-grained language-independent concepts. A set of near-synonyms is simply a set of words that all link to the same language-independent concept. So, in effect, near-synonyms share the same propositional meaning (or contribute to the truth-conditions in the same manner) up to a point in granularity defined by language-dependence. Thus, if we are designing a lexical resource for use in multilingual applications, we have an operational definition of near-synonymy: if the same concept has several reasonable lexicalizations in different languages then it is a good candidate for a language-independent concept, its various lexicalizations forming sets of near-synonyms in each language. Of course, in a unilingual application, we would have to come up with some other, perhaps domain-dependent, cut-off point in granularity (though the multilingual principle could also be used in such an application.)

2.4 A definition of near-synonymy

In conclusion, it appears that it might not be possible to precisely and rigorously define near-synonymy. The definition might depend to a large extent on how the near-synonyms are to be used, that is, on the type of system, whether for word-understanding, word-finding, or word-choosing, and whether the application is multilingual or unilingual. In the end, for a computational system for lexical choice, we probably don't need a fully rigorous definition. So, in this research, I'll use the following definition:

> *Near-synonyms* are words that are alike in essential (language-neutral) meaning, or denotation, but possibly different in terms of only peripheral traits, whatever they may be.

A slightly more rigorous definition, explicitly involving the idea of granularity, is that

> *Near-synonyms* preserve truth-conditions, or propositional meaning, to a level of granularity of representation consistent with language-independence in most contexts when interchanged.

The existing synonym dictionaries will supply the sets of near-synonyms. This definition of course doesn't resolve the debate on synonymy, but that debate is what lexical semantics is all about, so it will no doubt go on indefinitely. The definition does however resolve the synonymy paradox, so it does have theoretical implications for lexical semantics. (In chapter 5 we will explore these implications further in fleshing out a theory for representing fine-grained lexical knowledge that involves clusters of near-synonyms.)

Chapter 3

A classification of near-synonymic differences

Recognizing merely that near-synonyms are similar in meaning is insufficient if we are to develop a system for lexical choice—we must also represent their *differences*. Clearly, near-synonyms can differ in many ways, but unfortunately, no thorough computationally-motivated classification of near-synonymic variation yet exists. So, in this chapter we present such a classification that is the most comprehensive yet, with 35 types of variation distinguished. But still, it is only a step towards an ultimate classification of the phenomena that would have to be accounted for in the most sophisticated lexicon.

First, a few words about our methodology. In building a classification one should strive to be objective. There are essentially three sources from which to find types of difference, listed here in decreasing order of the extent to which one might suspect the source to reflect one's own biases.

1. From one's own intuition.

2. From classifications and theories proposed by other linguists, lexicographers, semanticists, and researchers.

3. From dictionaries (i.e., from lexicographers) whose purpose it is to discriminate near-synonyms.

The first source is, of course, the most direct but also the most subjective, but sometimes one's intuition is an important factor. The second source can also involve personal bias, but combining the results of many researchers might allow us to be more objective. Of course, the types of differences that researchers list may be based on their intuition, and may be biased towards a particular theory. Often, the types that they list are the result of their investigating the third source. The third source would seem to be the most fruitful, but it is less direct, since one must generalize from the specific differences described in the dictionaries. And although lexicographers are also subject to biases, they are at least trained to be as objective as possible: Gove (1973, p. 25a) claims that "we have always tried to base our judgment upon evidence that was not affected by any personal prejudices or predilections".

Actually, there is a fourth source: raw text corpora. That is, we can directly analyze a large corpus of text to determine how words are used, and thereby determine differences in meaning and usage. Of course, this is what lexicographers do—by hand—by using a combination of objective methods, experience, and intuition. But the work is so labour-intensive that it would be beneficial to devise automatic methods to do the same thing, or, in the least, to assist lexicographers. So far, though, automatic methods involve relatively simple statistical analysis, so we must still rely for the most part on the already-produced reference books.

Our methodology was to use the second source to build a broad classification (relying on Ullmann (1962), Cruse (1986), Hovy (1988), DiMarco, Hirst and Stede (1993), Stede (1993), and others), and to refine it, especially for denotational variation, by analyzing the content and form of synonym usage notes in various synonym dictionaries (Fernald, 1947; Gove, 1973; Room, 1981; Hayakawa, 1994; Summers, 1993; Bailly, 1947; Bénac, 1956; Batchelor and Offord, 1993).

In a separate line of enquiry, we automatically analyzed a large text corpus to extract statistical differences in the choice of near-synonyms in context; this statistical method will be taken up in chapter 6. The remainder of this chapter is devoted to the knowledge-based classification.

Finally, we investigated differences in only open-class words, that is, verbs, nouns, and adjectives. And, for each type of variation in the classification, we try to give 'pure' examples (i.e., words that differ in only the one aspect), but this is not always possible since words usually differ on many dimensions at once.

3.1 Broad classification

As one would expect, near-synonyms can differ according to any aspect of their meaning. Cruse (1986) breaks the range of variation into four broad categories of meaning, listed here:

- Propositional, or denotational, meaning.
- Expressive meaning, including affect, emotion, and attitude.
- Stylistic meaning, including dialect and register.
- Presupposed meaning, leading to selectional and collocational restrictions.[1]

The classic opposition of *denotation* and *connotation* does not appear to be precise enough. 'Denotation' refers to the literal, explicit, and context-independent definition of a word, whereas 'connotation' refers to any aspect that is *not* denotational including ideas that colour a word's meaning, emotions, expressed attitudes, implications, tone, and style. 'Connotation' is simply too broad and ambiguous a term. It needlessly conflates aspects of the middle two categories above—though the line between them is fine to begin with. 'Connotation' is also often used to refer to aspects of a word's meaning that, while being denotational in nature, are either fuzzy or peripheral to the meaning of the word and are often expressed only indirectly depending on the context.[2] Although Cruse discusses this type of meaning in the context of near-synonymy, it's not clear into which of his categories it falls, if at all.

Gove (1973) explicitly includes peripheral meaning in his breakdown, which is otherwise similar to Cruse's, except that it is less precise, being the result of lexicographical practice rather than theoretical semantics. According to Gove (p. 25a), synonyms may have distinctions in:

[1] Cruse's use of the expression "presupposed meaning" is not related to the "presupposition" of pragmatics. He explains that it is a pre-theoretical notion for the semantic traits that are "taken for granted in the use of a lexical item" (p. 278), referring to what are usually called selectional restrictions. He also uses the expression "evoked meaning" to refer to the effects of using words from another dialect or register.

[2] It seems that 'connotation' is used to refer to any aspect of word meaning that we don't yet understand well enough to formalize. See also the discussion in section 1.2.

> (1) implications—here, mostly minor ideas involved in the meaning of the word; (2) connotations—the ideas which color the meaning and are the product of various influences, such as etymology, language of origin, and historical or literary association; (3) applications—the restrictions in a word's use as prescribed by idiom or in accordance with the nature of the other words with which it may be associated.

Gove's implications cover both propositional meaning and peripheral meaning. His connotations would seem to coincide with Cruse's stylistic meaning (at least dialect), and perhaps aspects of Cruse's expressive meaning. And his applications coincide with Cruse's presupposed meaning. Strangely, Gove does not explicitly include stylistic meaning even though many of his usage notes discuss distinctions in the formality of words.

For our top-level broad classification, we will use a combination the above two classifications. The differentiation of near-synonyms can involve:

- collocational and syntactic variation,
- stylistic variation (including dialect and register),
- expressive variation (including emotive and attitudinal aspects), and
- denotational variation (in a broad sense, including propositional, fuzzy, and peripheral aspects).

The next four sections further discuss and subdivide each of these broad categories. We then discuss variation across different languages, and finally we give an example of multidimensional variation.

3.2 Collocational and syntactic variation

Near-synonyms can differ in how they combine with other words in their surrounding local context. These co-occurrence patterns range from predictable *selectional restrictions* to *lexical collocations* and *grammatical collocations* to *idioms*. Some examples are shown in table 3.1.

Selectional restrictions are generally defined to be a direct consequence of the denotation of words. For instance, the agent of *eat* must be an animate creature, not an inanimate object, because of the denotation of *eat*.[3] Because selectional restrictions are denotational, words that would otherwise be near-synonyms, are sometimes not considered so. For instance, while anything that can fly, including airplanes, can *land* or *alight*, only birds can *perch*. Similarly, while a person or animal can *eat*, only a person can *dine* (in non-metaphorical use).

According to Benson (1990), a collocation is an arbitrary and recurrent word combination. But some collocations are not as arbitrary as others. For instance, Cruse (1986) claims that a *customer* typically acquires something physical, whereas a *client* typically receives a less tangible service; thus bakers, shoe-shops, and stores have customers, while architects and lawyers have clients (however, banks have customers and clients, so the rule is not completely systematic, which is exactly the point). Similarly, vegetables and meat can go *rotten*, but not *rancid*, because *rancid* usually applies to decomposing oils. A fully arbitrary collocation is perhaps

[3] One can also think of selectional restrictions as one end of a continuum of co-occurrence patterns if classes of words are taken into account rather than just lexical items (Resnik, 1993). But it is inescapable that selectional restrictions involve semantics, because otherwise we would not be able to understand expressions that break them: e.g., *The vacuum cleaner ate my diamond ring.*

Selectional restrictions	*die* : *pass away*, *land* : *perch*, *eat* : *dine*
Lexical collocation	*customer* : *client*, *addled* : *rancid* : *rotten*, *task* : *job* w.r.t. *daunting*
Grammatical collocation	*correct* : *right* w.r.t. *a*|*the*
Idiom	*bite the dust* : **gnaw the powder*
Subcategorization	*give* : *donate*, *teach* : *instruct*
Converse	*like* : *please*, *buy* : *sell*

Table 3.1: Collocational variation.

daunting task; *daunting job* certainly sounds odd, even though its meaning is clear. Also, some collocations involve closed-class lexical items, and thus might be better called ‘grammatical collocations’. For instance, one says *the right answer* but not *a right answer*, yet *correct* can be used in both ways.

At the end of the scale are idioms, which are so idiosyncratic that they admit no intersubstitution. For instance, we cannot replace any word of *bite the dust* and have it make sense: *gnaw the powder?*

Some near-synonyms differ in the way they assign grammatical roles (e.g., subject or object) to their semantic roles. For instance, in 3.1a, *give* places the beneficiary in the indirect object position, but *donate*, in 3.1b, does not allow this. Example 3.2 shows that a similar pattern exists for *teach* and *instruct*, but in a different syntactic frame.

3.1a. Zach gave the museum his collection of lobster claws.

b. *Zach donated the museum his collection of lobster claws.

3.2a. Zach teaches lobster trapping to tourists.

b. *Zach instructs lobster trapping to tourists.

Lexical converses, considered a form of antonymy by some, are actually more like synonyms because they convey the same denotational meaning but from a different perspective. Treated as synonyms, they are similar to the subcategorization examples. Consider examples 3.3 and 3.4. The verbs simply assign different grammatical roles to their participants.

3.3a. Zach likes seafood.

b. Seafood pleases Zach.

3.4a. Sam sold lobster to Zach.

b. Zach bought lobster from Sam.

Geographical dialect	*loch* : *lake*, *chips* : *fries*, *colour* : *color*, *booze can* : *after-hours club*
Temporal dialect	*lapidate* : *stone (to death)*, *mundivagant* : *globetrotting*, *wireless* : *radio*, *color* : *colorize*
Social dialect	*loo* : *toilet*, *daddy* : *father*, *tude* : *attitude*
Language	*verboten* : *forbidden*, *c'est la vie* : *it is fated*
Sublanguage	*brace [cricket]* : *two*, *matrimony [religious]* : *wedlock [legal]* : *marriage*

Table 3.2: Dialectal variation.

3.3 Stylistic variation

Style is an essential part of meaning, and is "created through the subtle variation of what is said, the words used to say it, and the syntactic constructions employed" (DiMarco and Hirst, 1993, p. 452). So, where the lexicon is concerned, stylistic variation involves tuning the surface form, rather than the denotation, of the words used to express an idea. We will subdivide style into two areas: *dialect* and *tone*, though the division is somewhat fuzzy.

3.3.1 Dialect and sublanguage

Dialects and sublanguages are associated primarily with either a group of language users or a certain domain of language use. Cruse (1986) subdivides dialect three ways into geographical, temporal, and social dialects. Table 3.2 shows some examples of dialectal and sublanguage variation.

Geographical dialects assign a different word (or spelling) to the same thing. However, the words are not exactly interchangeable. A Canadian speaking of a loch in Scotland would be incorrect to call it a *lake*, and a Canadian lake should not be called a *loch*. But, the non-canonical word could be chosen for effect. Temporal dialects are associated with the currency of words as they range from obsolete and archaic words, to established words, and to very recent, even trendy, coinages. It should not be confused with the common / rare distinction, which has to do with frequency of usage. Finally, social dialects are associated with identifiable classes of language users, whether they are social classes (e.g., lower and upper class), age groups, or classes aligned to specific ideals and morals. The use of slang is often associated with a social dialect, but it can also be classified on the dimension of formality (see below), or even as a separate dimension.

A whole other language can be thought of as a dialect from which to borrow words. When a borrowed word is used in place of a native word, it can be considered a near-synonym, and

Dimension	Suggested by
Formality	Hovy (1988), Nirenburg and Defrise (1992), Stede (1993), DiMarco, Hirst and Stede (1993)
Force	Hovy (1988), Nirenburg and Defrise (1992), Stede (1993), DiMarco, Hirst and Stede (1993)
Concreteness	Stede (1993), DiMarco, Hirst and Stede (1993), Toglia and Battig (1978)
Floridity	Hovy (1988), Stede (1993)
Colour	Hovy (1988), Nirenburg and Defrise (1992)
Simplicity	Hovy (1988), Nirenburg and Defrise (1992)
Familiarity	Toglia and Battig (1978)
Core or basic-level	Stede (1993), Reiter (1991)
Euphemism	Stede (1993), Allan and Burridge (1991)

Table 3.3: Various stylistic dimensions suggested in the literature.

might be chosen either because the native word is inadequate to convey the concept, or to create some other effect. I'll return to cross-linguistic synonyms in section 3.6.

Words are also used in special ways in sublanguages. A sublanguage is a distinct language, not necessarily a subset of general language, that is used in or created by a specific discipline or field. For example, lawyers, doctors, sports columnists, politicians, cooks, to name a few professions, all use sublanguages in which words take on special distinct meanings thereby creating sublanguage-specific near-synonyms. Examples of sublanguage, or technical, vocabulary are given in table 3.2, but this type of variation is virtually unconstrained.

Though one might consider building separate lexicons for each dialect and sublanguage, this would be inappropriate in many applications, since all of the options should be available at once. For instance, one can choose an archaic word for special effect in contemporary prose, or a word used by a different social class to evoke some meaning. Words "become alive with associations (i.e., evoked meaning) when transported to alien environments" (Cruse, 1986, p. 284).

3.3.2 Stylistic tone

Whereas dialectal variation is associated more with language users, tonal variation is associated with contexts or situations of language use, and its main purpose seems to be to provide ways of intentionally evoking a tone or modulating the existing tone of an utterance. Also unlike dialectal variation, types of tonal variation are better thought of as continuous dimensions of variation.

Formality is the most recognized dimension; consequently, many dictionaries mark formal and informal (or colloquial) variants. *Force* is fairly common as well, but it is not usually explicitly identified in dictionaries. Many other dimensions have been suggested in the literature, but there is little agreement on a representative set. Table 3.3 presents a few dimensions with the names of the researchers who proposed them. (Hovy (1988) used style manuals and his own intuition; Nirenburg and Defrise (1992) modified Hovy's; Stede (1993) and DiMarco,

Formality	*put up* : *construct* : *erect*, *flick* : *movie* : *motion picture*, *pissed* : *drunk* : *inebriated*, *kicked the bucket* : *died* : *passed away*
Force	*important* : *critical*, *refuse* : *repudiate*, *ruin* : *wreck* : *destroy* : *annihilate*
Concreteness	*name* : *christen*, *error* : *blunder*
Floridity	*house* : *habitation*, *request* : *solicitation*
Euphemism	*toilet* : *bathroom* : *washroom* : *restroom*, *fat* : *Rubenesque*, *Indian* : *native* : *person of indigenous background*, *masturbate* : *play with oneself* : *slap the monkey*
Familiarity	*smirk* : *smile*, *gaze* : *look*, *divulge* : *disclose* : *reveal* : *tell*
Simplicity	*hound* : *dog*

Table 3.4: Stylistic variation. All near-synonyms are presented in increasing order on the relevant dimension.

Hirst and Stede (1993) adduced dimensions from synonym usage notes; and Toglia and Battig's (1978) arose from psycholinguistic research.) The problem is that the dimensions cannot be defined in clear-cut terms because there are many correlations between them. For example, formal words are often flowery, and informal words can be slang words; forceful words are more intense or 'stronger' than others, so they are often simple, non-flowery, and concrete; concrete words, that is, words that are more apt to evoke a vivid mental image,[4] are sometimes more semantically specific than their near-synonyms; floridity and colour may be the same dimension: they refer to 'bookish', literary, or sophisticated words, so they are often more formal and less simple; simple words are often more familiar or common; core or basic-level words, while also involving a semantic phenomenon (Rosch, 1978), might correlate with simple words and with familiarity; finally, euphemistic words—words used in place of dispreferred or taboo alternatives in order to save face—are in some sense opposite to dysphemistic words—words used intentionally because of their offensiveness, but words at both ends of the scale are often colloquial and informal (Allan and Burridge, 1991). (Also, dysphemistic words can be used to express a negative attitude towards someone; see the next section.)

The complex interrelationships between the various stylistic (tonal) dimensions, and their connection to a speaker's communicative goals are not yet well understood. Many pragmatic factors are involved including the type of relationship between the speaker and hearer (or

[4] Psycholinguistic studies have shown a strong correlation between *concreteness* (the degree to which a word refers to a concrete object), *imagery* (the capacity of a word to arouse a mental image), and *categorizability* (the ease with which a word can be put into some larger category). A similar study showed that *familiarity* (how commonly a word is felt to have been experienced) strongly correlates with raw word frequency data (Toglia and Battig, 1978).

writer and reader), how the speaker wishes to change or affect the relationship, the setting of the communication, social and cultural conventions, and so on (Hovy, 1988). Whatever the dimensions are, the one thing that is clear about stylistic dimensions is that they are absolute dimensions in that every word in a language (and possibly across languages) can be compared to every other word on the same finite set of dimensions. This is because each dimension would appear to arise out of certain situations of language use (i.e., words that are usually used in, say, formal situations and not usually in others, gain a degree of formality that carries over to other situations in which they are used).

So, although we cannot provide a definitive set of dimensions now, we can list a few that seem more important or useful in language use (see table 3.4) and assume that they, and others, should be representable in the same formalism in the lexicon. As mentioned above, these are continuous dimensions with a range of possible values from low to high; some may even have a neutral position in the middle—the dimension of 'formality', for instance.[5]

3.4 Expressive variation

Words can convey feelings or emotions and also opinions or attitudes, both of which are forms of expressive meaning. However, we have to be careful because many emotions are expressed merely by the denotation of the word (e.g., with words like *hate, cheerful,* or *surprise* : *astonish* : *amaze* : *astound*). Cruse distinguishes expressive and denotational meaning in terms of truth-conditions; only the latter plays a role in determining the truth-conditions. So, the purpose of varying expressive meaning must be to express one's feelings or value-judgments in an indirect way, or to arouse emotions in another. Thus, unlike style and denotation, expressive variants explicitly involve the speaker and sometimes the hearer, who are often 'external' to the situation that the words are describing.

There are two main categories of expressive variation: words that express varying emotions of the speaker, and words that express the speaker's attitudes or value-judgments of a participant in the situation. Table 3.5 gives some examples. Along emotive lines, Toglia and Battig (1978) report on a study in which human subjects ranked the *pleasantness* of words, but other feelings or emotions can also be expressed indirectly by words. For instance, *teeny-weeny* expresses amusement, and *tiny* can express surprise (Gove, 1973; Hayakawa, 1994); *daddy* expresses a stronger feeling of intimacy than *father*; *reject* is also stronger emotionally than *decline* (Hayakawa, 1994); and *goof* has a humorous tone compared to *error*. This kind of variation could be represented in much the same way as stylistic variation, but one would have to identify all of the possible dimensions of emotion (Davitz, 1969).

On the other hand, variations in expressed attitude, because they explicitly involve a participant of the situation, require a more complex representation. Words can be pejorative, neutral, or favourable, though the dimension is continuous, and the participant of whom the opinion is expressed varies depending on the part of speech of the word. Adjectives usually express an attitude towards the participant that they modify (e.g., *skinny* : *slim*); nouns sometimes towards the participant to which they refer (e.g., *banger* : *car*), but sometimes towards another participant (e.g., *blunder* : *error*); and verbs usually towards their agent (e.g., *blubber* : *cry*).

[5] Stede (1993) encodes 'concreteness' and 'floridity' with a neutral position as well.

Emotive	*teeny-weeny* : *tiny* : *miniature*, *father* : *daddy*, *reject* : *decline*, *goof* : *error*,
Expressed attitude	*pushy* : *aggressive*, *skinny* : *thin* : *slim*, *ponderous* : *heavy* : *weighty*, *banger* : *car*, *blunder* : *error*, *blubber* : *cry* : *wail*, *impute* : *attribute* : *credit*

Table 3.5: Expressive variation.

Technical distinctions	*alligator* : *crocodile*, *rabbit* : *hare*, *sleet* : *hail*, *pig* : *hog*, *slander* : *libel*

Table 3.6: Explicit fine-grained technical variation.

3.5 Denotational variation

Denotational, or semantic, variation appears to be the most complex type of variation to sort out. Not only can near-synonyms differ with respect to any concept or idea, but they can also differ in how the idea is conveyed (e.g., in terms of emphasis, necessity, and / or strength). That is, a concept conveyed by a word need not be expressed explicitly in every possible context; it may be merely implied or suggested in particular contexts, or it may be emphasized relative to another concept. So, a concept that is said to be denoted by a word in the traditional sense of 'denotation' is not, in practice, always a necessary implication of using that word. Additionally, classifying the types of semantic variation is difficult because lexicographers do not frame usage notes in terms of explicit types of variation; they simply explain, in regular language, specific differences between near-synonyms.[6] We now discuss several types of semantic variation.

3.5.1 Fine-grained technical distinctions

To start, many ostensible near-synonyms have explicit fine-grained technical distinctions that actually make them merely similar or related words (see table 3.6). However, if one is unaware,

[6] Of course, one would not expect lexicographers to develop a comprehensive theory of lexical differentiation (though it would no doubt help in their endeavours), but merely to determine how particular sets of synonyms differ.

Intentional / Accidental	*stare at* : *glimpse*
Continuous / Intermittent	*seep* : *drip*
Immediate / Iterative	*strike* : *beat*
Emotional / Non-emotional	*relationship* : *acquaintance*
Degree	*mist* : *fog*

Table 3.7: Abstract denotational variation (DiMarco, Hirst and Stede, 1993).

unclear, or unconcerned about the distinctions, then they could be considered synonyms. For instance, to the layman the difference between an *alligator* and a *crocodile* might be irrelevant, but not so to the biologist.[7] In this case, the distinction is determined primarily by the nature of the referents involved; other examples include *rabbit* : *hare* and *sleet* : *hail*. In other cases, the distinction may be 'legislated'. For example, the difference between a *hog* and *pig* under the U.S. meat classification system is that a *hog* is an adult swine weighing more than 120 pounds when ready for market, and a *pig* is an immature swine weighing less (Randall, 1991). Similarly, *slander*, in legal use, refers to a defamation spoken in the presence of others, whereas *libel* refers to a defamation put in writing.

In all of these examples, the distinctions are clear-cut, either by nature or by definition, so they can often be represented by simple features specific to the words being distinguished. But even though the words overlap significantly in meaning, they are not really near-synonyms at all, because the distinctions are clearly intended as contrasts in meaning. (Recall Cruse's observation in 2.2 that synonyms usually exhibit a low degree of contrastiveness.) The pairs aren't antonyms either, but the relation seems analagous to antonymy. However, the words often have less technical, but closely related, senses that have less clear-cut distinctions. For example, in general usage the implications of *slander* and *libel* are much the same (Gove, 1973).

3.5.2 Abstract features

Some types of semantic difference are broadly applicable to many groups of synonyms, while some are specific to only a single group. DiMarco, Hirst and Stede (1993) attempted to determine differences of the former type in their study of usage notes.[8] They found 26 denotational dimensions, both binary and continuous, that were used quite frequently. The examples they give involve mostly basic aspects of verb semantics; however, they also adduce dimensions of emotion and degree; see table 3.7. It appears then that many abstract differences can be characterized as differences in the possession or not of a simple feature, or in the position on a continuous dimension. However, DiMarco, Hirst and Stede also found that many (more specific) sorts of distinction could not be characterized in this manner, but they did not undertake a detailed analysis to determine how prevalent such differences are. For instance, some near-synonyms differ in the emphasis of a component of meaning (e.g., *enemy* : *foe*), and others have fuzzy overlapping meanings (e.g., *forest* : *woods*); these and other examples will be discussed in the next sections.

[7] In appearance, the alligator's snout is shorter and blunter than the crocodile's, and it shows no teeth when closed. The crocodile is considered more aggressive than the alligator. Also, their habitats differ (Randall, 1991).

[8] They used the *Oxford Advanced Learner's Dictionary (OALD)* and the *Longman Dictionary of Contemporary English (LDOCE)*, which together contain about 600 usage notes.

Prevent is the comprehensive term for the group. Used of human agency, it implies precautionary or restraining measures taken to hinder, thwart, or block a thing. *Prevent* is used of a nonhuman agency or chance cause that acts as a hindrance; in such cases, only the effect of the agent is stressed and no motive is attributed to it.

Stop in this sense is more informal than *prevent* and has greater immediacy. Where *prevent* often implies forethought and advance preparation to keep something from occurring, *stop* focuses on responsive action to the circumstance at hand.

To **preclude** is to *prevent* by anticipation or to rule out by necessity. *Preclude* is used not of persons, but of events, circumstances, decisions, or the like. It suggests a door shut in advance, as a consequence of a prior action, occurrence, choice, or commitment.

Forestall is like *preclude* in emphasizing anticipation, but it implies deliberate deterrence. Where *preclude* points to circumstances that make an occurrence impossible, *forestall* involves taking preventive countermeasures in time to turn something aside or temper its effect.

Avert, more strongly and more specifically than *forestall*, implies warding off a threat or danger. It points to the taking of effective countermeasures just in time to keep something disastrous from happening.

Obviate is the most formal verb in this set. It means dispose of, referring to a risk, difficulty, objection, or the like that is met squarely and cleared out of the way.

Figure 3.1: Hayakawa’s (1994) entry, abridged, for the near-synonyms of *prevent*. Terms used to describe differences are underlined.

3.5.3 Variation in the manner of expression

Consider the complexity of Hayakawa’s (1994) discrimination of the near-synonyms of *prevent* in figure 3.1. (Also consider Gove’s (1973) entry for *error*, in figure 1.1, on page 3.) The near-synonyms differ not only with respect to the expression of different concepts and ideas, but in the manner in which the concepts are expressed. In describing the manner, Hayakawa uses terminology such as “implies”, “stressed”, “focuses”, “used of”, “suggests”, “emphasizing”, “points to”, “referring”, and “involves”, as well as explicit comparisons (i.e., “more strongly than”, “is like”), and degree words such as “often”. Gove uses similar terminology, and a casual survey of both Hayakawa’s and Gove’s usage notes (and those in the other synonym dictionaries) shows that this form is reasonably consistent. For example, figure 3.2 compares Gove’s and Hayakawa’s entries for the near-synonyms of *lie* ‘untrue statement’; not only do the notes agree for the most part in content, but they also contain similar terminology.

In order to be certain of this, we analyzed in detail the form and language of synonym usage notes in *Webster’s New Dictionary of Synonyms* (Gove, 1973) and in *Choose the Right Word* (Hayakawa, 1994). We studied the first 99 usage notes of the former, which comprise 607 separate distinctions, and the first 53 usage notes of the latter comprising 385 distinctions. Table 3.8

Gove:

Lie is usually felt to be a term of extreme opprobrium because it implies a flat and unquestioned contradiction of the truth and deliberate intent to deceive or mislead.

Falsehood may be both less censorious than *lie* and wider in its range of application. ... Like *lie*, the term implies known nonconformity to the truth, but unlike *lie*, it does not invariably suggest a desire to pass off as true something known to be untrue.

Untruth is often euphemistic for *lie* or *falsehood* and may carry similar derogatory implications. ... Sometimes, however, *untruth* may apply to an untrue statement made as a result of ignorance or a misconception of the truth.

Fib is an informal or childish term for a trivial falsehood; it is often applied to one told to save one's own or another's face.

Misrepresentation applies to a misleading and usually an intentionally or deliberately misleading statement which gives an impression that is contrary to the truth.

Hayakawa:

Lie is the most general of these, but is restricted to a conscious and deliberate intention to distort the truth.

Fib, the most informal, is exclusively used [as a euphemism for *lie*], suggesting a trivial, harmless, or forgivable *lie*.

Prevarication, the most formal of the nouns here, would be taken by many as an overly fancy euphemism for *lie*. ... *Prevarication* can have a special area of meaning that refers not to bald misstatements of fact but to the deceptive statement of half-truths, but this distinction would be lost on most people.

Falsehood and **untruth,** as euphemisms, are less formal circumlocutions than *prevarication*. *Falsehood*, however, has a legitimate reference to any incorrectness, whether intentional or not. *Untruth* can sometimes refer to fictions that were never intended to mislead or be taken as fact.

Figure 3.2: A comparison of Gove's (1973) and Hayakawa's (1994) entries (both abridged) for the near-synonyms of *lie*.

Category	*Webster's*	*Right Word*
Denotation	84	209
Implication	249	45
Suggestion	123	76
Connotation	53	9
Presupposition	7	1
Stress	54	19
Emphasis	12	15
Application	25	11
Total	607	385

Table 3.8: Categories of semantic distinctions made in 99 usage notes of *Webster's New Dictionary of Synonyms* and 53 usage notes of *Choose the Right Word*.

breaks the semantic distinctions down into eight categories according to the basic terminology used.[9]

First, while the two editors do use similar terminology, there are distributional differences, with Hayakawa clearly favouring denotational distinctions over any other kind. The differences between the two books show a difference in methodology—Hayakawa is less precise and usually describes what words can refer to, whereas Gove and his lexicographers are more concerned with the implications of word use. Hayakawa also tends to discuss the different but related senses of words at the same time as he discusses differences between near-synonyms, thereby confounding his usage notes with words that would not normally be considered near-synonyms.

Now, the meaning of these eight terms is far from clear. For instance, 'implication' is ambiguous between a sense of 'logical necessity' (i.e., denotation), and a sense of 'indirect indication'. The words *suggest* and *imply* are themselves differentiated in *Webster's New Dictionary of Synonyms*: "very often the difference between *suggest* and *imply* is not clear, though *suggest* often connotes the necessity of delicate perception while *imply* connotes the need of inference". 'Connotation' is very ambiguous in both books. Gove says *connote* and *imply* / *suggest* are analogous words (i.e., not quite synonyms) and Hayakawa says, in his usage note for *connote*, that it "refers to the possible associations implied or suggested by a term [. . . and is] closer in meaning to *imply* than *suggest*." But both editors sometimes also use *connote* to describe differences in overtones and attitudes conveyed by words. 'Presupposition', as it is used here, also seems to coincide with implication in its sense of indirect indication. As these first five terms are similar, we could possibly conflate them into a single term (i.e., a single type of difference), but we are reluctant to throw away the information that we presume the lexicographers have carefully

[9] For each semantic distinction in a usage note, we determined into which category it belonged based on the terminology used by the lexicographer. For instance, we took any of the following terms to describe a denotational distinction: 'is', 'denote', 'mean', 'meaning', 'meaningful', 'apply', 'refer', 'reference', 'regarded', 'describe', 'signify', 'designate'; any of the following to describe an implication: 'imply', 'implication'; any of the following to describe a suggestion: 'suggest', 'suggestion', 'hint'; and so on. However, some terms were ambiguous. For instance, 'indicate' sometimes described a denotation and sometimes a suggestion; and 'apply' and 'refer' sometimes described an application rather than a denotation. In such cases, we used our intuition to determine the proper category.

Denotation	*account* : *chronicle* : *report*, *absurd* : *preposterous* : *ridiculous*,
Implication	*accompany* : *attend* : *escort* : *convoy*, *error* : *blunder* : *mistake* : *slip* : *lapse*, *lie* : *fib* : *untruth* : *misrepresentation*, *casual* : *fortuitous* : *contingent*
Suggestion	*snoopy* : *nosy* : *prying*, *help* : *aid* : *assist*, *acknowledge* : *admit*, *adjust* : *adapt*

Table 3.9: Variation of indirectness of expression.

and deliberately recorded. However, we can generalize from these terms: there appears to be a continuum of *indirectness* of expression from 'denotation' to 'implication' to 'suggestion'. We discuss this continuum below.

'Stress' and 'emphasis', and their variants, are used interchangeably by lexicographers to refer to the idea of making some aspect of the meaning of a word more prominent. But sometimes it appears that 'emphasis' is used to mean 'implication' or 'suggestion', rather than true emphasis (e.g., in the usage note for the group of *prevent*-synonyms, Hayakawa says that *forestall* is "like *preclude* in emphasizing anticipation", yet he also seems to say that *preclude* denotes *anticipation*), and sometimes 'implies' is used as 'emphasis': "*absorb* may imply the thoroughness of the actions. ... *digest* is like *absorb* in stressing thoroughness, but is even more emphatic. ... *assimilate* is even more emphatic about the thoroughness ... " (Hayakawa, 1994, p. 1). We will return to emphasis below.

'Application' refers to the restrictions in how a word combines with other words, that is, to its selectional restrictions and collocational patterns. This type of difference was covered in section 3.2, above.

Indirect expression of meaning The continuum of indirectness introduced above might equally be called one of necessity (see Cruse's (1986) definitions of *criterial*, *expected*, and *possible* traits), or of strength of implication. In fact, the strength of an implication (or denotation, or suggestion) is sometimes explicitly stated in the usage note (see *slip* in figure 1.1). So, while denotational distinctions are logically entailed by a word, its implications and suggestions are mere possibilities that a reader has the license to decide were not intended to be expressed. Hence, they cannot be modelled as necessary and sufficient truth-conditions of the word. How indirect a writer chooses to be will of course have pragmatic ramifications on the discourse. For instance, the more indirect an expression of some fact is, the less likely the reader is to infer the fact and pay attention to it, and the more easily it can be cancelled by a separate statement by the writer.

Table 3.9 shows some examples of near-synonyms that differ in the indirectness with which they express certain concepts. Sometimes, the difference is described explicitly in the usage note by a comparison between the words, but most differences of this sort are only implicit by virtue of the words being compared in the same usage note.

Emphasis rel. to other concept	*scurrility* : *abuse*, *degrade* : *abase*, *lapse* : *slip*
Emphasis rel. to other word	*enemy* : *foe*, *absorb* : *digest* : *assimilate*, *provide* : *supply* : *furnish*

Table 3.10: Variation in emphasis.

Under denotation, whereas *account*, *report*, and *chronicle* all denote an oral or written statement, *report* also denotes that the statement is about something witnessed, and *chronicle* that it is detailed and extended (Gove, 1973). Similarly, *absurd* denotes that something is opposed to reason or truth, but *preposterous* adds that it is outrageously contrary to reason, and *ridiculous* that it invites ridicule (Hayakawa, 1994).

Under implication, *accompany* implies equal status, whereas *attend* implies subordinate status (of the attending person); *escort* and *convoy* add to *accompany* the implication of protection. *Mistake* carries a weaker implication of inadvertence or accident than *slip*, which also implies triviality; *lapse* has a weaker implication of triviality than *slip*. All five *error* words differ in their implication of severity of criticism. The words for *lie* differ in their implications of deliberateness and directness; *lie*, *fib*, and *misrepresentation* all imply a deliberate attempt to mislead, but *untruth* and *falsehood* can imply an untrue statement made unintentionally or through a misconception. *Fibs* are trivial and often face-saving, whereas a *lie* is direct, and a *misrepresentation* can be indirect. Finally, *casual*, *fortuitous*, and *contingent* imply chance at different strengths (Gove, 1973).

Under suggestion, *snoopy* and *nosy* can suggest an avidity for spying on others or gathering gossip, the former suggesting stealth, the latter more forthright methods; *prying* goes further and suggests an absolute invasion of privacy by any means whatever (Hayakawa, 1994). Compared to *help*, *aid* strongly suggests the need for assistance; and compared to *aid*, *assist* suggests a secondary role in the assistant. *Admit* has a weaker suggestion of disclosing something that has been concealed than *acknowledge*. Finally, in contrast with *adapt*, *adjust* suggests less flexibility and more ingenuity (Gove, 1973).

Emphasis Emphasis is inherently relative, but there are two kinds of relativity used in the usage notes (see table 3.10). First, a word can emphasize (or foreground) an idea relative to another idea (that is backgrounded). For instance, *scurrility* ‘vehemently expressed condemnation’ emphasizes the quality of the abuse rather than the attack itself; and *degrade* ‘lower in estimation’ emphasizes a lowering in plane rather than in rank (Gove, 1973). Of course, in these two cases the emphasis is implicitly relative to their near-synonyms (*abuse* and *invective* for the former, and *abase* and *demean* for the latter), but it is not clear that any of these near-synonyms actually emphasize, or even imply, the ideas backgrounded by *scurrility* and *abase*, or de-emphasize the foregrounded ideas.

Second, a word can emphasize a particular idea relative (explicitly) to another word. In these cases, it is not necessary for another idea to be backgrounded or de-emphasized. For instance, *enemy* emphasizes (or stresses) antagonism or hatred, an emotional reaction, whereas

	Category				
Frequency term	Denotation	Implication	Suggestion	Connotation	Stress
always	1	5	1	0	0
commonly	0	6	3	2	1
frequently	1	6	4	4	1
may	7	10	3	4	1
often	2	25	18	12	5
sometimes	2	5	10	4	3
usually	2	25	4	2	3
Other	8	30	14	8	3
Total modified	23	112	57	36	17
Not modified	61	137	66	17	49

Table 3.11: Frequency terms used to modify five types of distinctions in *Webster's New Dictionary of Synonyms*. The 'Other' row accounts for 30 other frequency terms. The 'Not modified' row lists the total number of distinctions of each type that were not modified at all by a frequency term.

foe does not, implying instead active warfare (Gove, 1973). And *absorb, digest,* and *assimilate* 'to take in, especially mentally' emphasize thoroughness in progressively stronger degrees (Hayakawa, 1994). There are also more complex cases of emphasis that combine the two types: the differences between *provide, supply,* and *furnish* are that they stress, respectively, preparing for something by stocking or equipping, replacing or making up what is needed, and fitting someone with whatever is necessary (Gove, 1973).

Frequency of expression More than a third of the distinctions (not including applications) that are summarized in table 3.8 are qualified by a frequency term (245 out of 575 in *Webster's* and 101 out of 374 in *Choose the Right Word*). Table 3.11 lists some of the terms used. Frequency terms represent the frequency with which a synonym expresses the given concept in the corpora that the lexicographers used. But whether or not a word has expressed or will express a given concept in a particular context is very difficult to resolve. (Frequency terms are a means to gloss over the complexity of the problem, but the lexicographers also provide exemplars of usage to help dictionary users in this respect.) Table 3.11 shows how pervasive frequency terms are across all categories of expression. Thus, it's clear that the context plays a large role in determining whether or not a particular difference is relevant. But it is much less clear how to classify the various influences that context can have. This leaves a hole in the classification and an open question for future research.

Table 3.12 shows a few examples of variation in frequency of expression. *Story* sometimes, and *version* always, implies contrast with another statement of the same events. *Blunder* sometimes implies blameworthiness, whereas *mistake* doesn't always, and *error* almost always does (at least this is implied by the usage note) (Gove, 1973). And of *retract* and *recant* 'abandon irrevocably', the former often implies an admission of error, but the latter always does.

Frequency of expression	*story* : *version*, *blunder* : *mistake* : *error*, *retract* : *recant*

Table 3.12: Variation in frequency of expression of an idea.

3.5.4 Dimensions of denotational variation

So far we have analyzed only the manner in which near-synonyms differ in expressing a particular concept, but the examples so far make it clear that any concept can be involved in such a distinction. While in sections 3.5.1 and 3.5.2, we saw that near-synonyms can differ with respect to fine-grained technical details, and with respect to abstract semantic features (or dimensions), here we will consider a few types of semantic variation that seem to involve full-fledged concepts rather than simple features. Table 3.13 summarizes the differences discussed below.

Basic dimensions Many concepts or ideas in which near-synonyms differ can be considered to be dimensions of variation. Some are obviously continuous dimensions (e.g., *error, mistake,* and *blunder* differ in degree of severity of criticism); some are binary dimensions having two opposite values (e.g., *escort* implies protection while *accompany* does not); and some, only more loosely called dimensions, involve multiple mutually exclusive values (e.g., *order* and *command* differ in terms of the authority of the agent: the former implies peremptory authority, the latter official authority).

Complex 'dimensions' Sometimes the concepts in which two near-synonyms differ are more complex to represent than are the basic dimensions above. For instance, near-synonyms can refer to different aspects of a process over time. For example, *start* suggests a mere setting out from a particular point, whereas *begin* suggests the taking of the first step of a process in fulfillment of a purpose, and *initiate* suggests reference to the first step with no implication of an ending. And, *obsolete* and *obsolescent* refer to different moments in the same process, the former to what has already passed out of use, the latter to what is now passing away (Hayakawa, 1994).

Differences that make reference to world-knowledge (e.g., to an idea of naturalness or properness) are also complex; so *error* implies a straying from a proper course and suggests guilt that may lie in failure to take proper advantage of a guide, whereas *mistake* implies a wrong possibly caused by a misconception. And *arrange* implies a placing of things into the proper or right relationship, whereas *organize* implies the things are put into a functional arrangement (Gove, 1973). In each case knowledge of the "proper course" or "proper relationship" is presumed.

Some concepts in which synonyms differ are related to one another by inference. Sometimes a relationship is either explicitly stated in the usage notes, but often it is implicit. For instance, if *antagonistic* 'so opposed as to cause interference' suggests mutual opposition, then it implies incompatibility and irreconcilability. And *adverse*, a near-synonym of *antagonistic*, conveys so strongly the idea of unfavourable opposition that it often means harmful or fatal (Gove, 1973). And implicitly, the blameworthiness that is only sometimes conveyed by *mistake* and always by *error* must be related to the fact that a *mistake* is usually caused by a misconception or

Continuous dimension	*mistake* : *error* : *blunder* [severity], *absorb* : *digest* : *assimilate* [slowness]
Binary dimension	*escort* : *accompany* [protection], *abandon* : *forsake* [renunciation], *blunder* : *mistake* [stupidity]
Multi-value dimension	*order* : *command* [authority], *accompany* : *attend* [status], *lie* : *misrepresentation* [contradiction], *adore* : *worship* : *idolize* [admiration]
Complex (process)	*begin* : *start* : *initiate*, *obsolete* : *obsolescent*
Complex (world-knowledge)	*error* : *mistake*, *arrange* : *organize*
Complex (inference)	*antagonistic* : *adverse*, *error* : *mistake*
Specificity	*eat* : *consume* : *devour* : *dine* : *gobble*,† *act* : *work* : *operate* : *function*, *forbid* : *ban* : *outlaw*, *accuse* : *charge* : *incriminate* : *indict* : *impeach*
Extensional overlap	*error* : *mistake* : *blunder*, *review* : *article*, *brainy* : *cunning*, *high* : *tall*
Fuzzy overlap	*mistake* : *error* : *blunder*, *forest* : *woods*, *marsh* : *swamp* : *fen* : *morass*, *amicable* : *neighbourly* : *friendly*

†The first term is the most general term.

Table 3.13: Dimensions of denotational variation.

misunderstanding (i.e., unintentionally).

Of course, these complexities, and others such as references to beliefs and intentions and references to spatial relationships—which are, admittedly, hardly dimensions at all—have been notoriously difficult to characterize and formalize in semantics. Therefore, the classification presented here into a category of complex 'dimensions' is only a crude beginning.

Specificity Frequently, one near-synonym of a group is more general than the others in that it has a broader range of meaning and thereby encompasses the meanings of the more specific words. *Eat*, for instance, is the most general term of the group that includes *consume, devour, dine, gobble, gorge, sup,* and *wolf*: any particular action of eating that can be referred to by a word from this list, can also be referred to as *eating*, but not vice versa. Other examples are shown in table 3.13. Sometimes the general word might be so general as to warrant isolating it as a hyponym of the more specific near-synonyms. For instance, perhaps *act* should really be represented as a hyponym of *work, operate,* and *function*. But just as often the more general word is too close in specificity, to be considered a true hyponym: e.g., *forbid* 'to refuse to allow' is a general term, but is still near-synonymous with *ban* and *outlaw*. In practice, it is difficult both to formally define the notions of specificity and closeness of specificity, and to apply these ideas in distinguishing synonyms; however, intuitions are often valid here.

Fuzzy and overlapping words Clearly, the meanings of a set of near-synonyms overlap to a great extent, when meaning is thought of as denotation; otherwise they would not be near-synonyms. But near-synonyms can also overlap in extension, that is, they can refer to the same instance of a thing in the world. For example, a particular instance of 'something that deviates from the norm' might be called any of *error, mistake, blunder, goof,* and so on, not because the instance is 'underspecified', but because each word is a different way of conveying almost the same thing—it is possible that the same thing (or event) can be called both an *error* and a *mistake*, but an *alligator* is never a *crocodile* (see section 3.5.1). Cruse (1986) calls this relation *micro-compatibility*[10] and gives as examples *review* : *article*, *brainy* : *cunning*, and *letter* : *note*. Taylor (1997) calls the relation *coextension*, after research on colour-naming vocabulary, and suggests that *high* and *tall* are coextensive. Many synonym groups exhibit some form of micro-compatibility because most of the semantic properties that distinguish synonyms are not strictly denotational, as we saw above. However, some do not overlap extensionally, such as the near-synonyms that differ in fine-grained technical traits (see section 3.5.1).

Near-synonyms may also overlap because their denotations are *fuzzy*. Any time a continuous dimension is involved in the denotation of a group of near-synonyms, fuzzy differences arise. For instance, *mistake, error,* and *blunder* differ in the degree of severity of criticism, but we cannot seriously say that *mistake* ranges from exactly 0.0–0.3 on the scale, and *error* from 0.3–0.6. It is more likely that *mistake* falls in a range of severity that overlaps with *error* and maybe even with *blunder*. Only an empirical investigation would tell.

Fuzzy differences can be quite complex. The distinction between a *forest* and a *woods* (or *wood*) is a complex combination of size, primitiveness, proximity to civilization, and wildness (as determined by the types of animals and plants therein), each of which is a continuous dimension (Room, 1981). And then there are other near-synonyms of *forest* (e.g., *copse, thicket, grove,* and *bush*), which also have fuzzy differences in size, type of tree, undergrowth, and so

[10] *Compatibility* is the lexical relation between items, such as *lawyer* and *accountant*, that have a common superordinate, here *profession*, and that are not mutually exclusive. Someone can be both a *lawyer* and an *accountant* (Cruse, 1986).

> **MISTAKE, ERROR Fehler** is a definite imperfection in a thing which ought not to be there. In this sense it translates both 'mistake' and 'error'. **Irrtum** corresponds to 'mistake' only in the sense of 'misunderstanding', 'misconception', 'mistaken judgment', i.e. which is confined to the mind, not embodied in something done or made. [Footnote:] **Versehen** is a petty mistake, an oversight, a slip due to inadvertence. **Mißgriff** and **Fehlgriff** are mistakes in doing a thing as the result of an error in judgment.

Figure 3.3: Farrell's (1977) usage note for the German near-synonyms of *mistake* and *error*.

on. Table 3.13 contains more examples of near-synonyms that have fuzzy differences.

3.6 Cross-linguistic variation

The above investigation used only English synonyms as examples. But of course, every language has its own synonyms, which can differ in the same ways as English synonyms do. But, since different languages need to be able to express the same meanings as each other, cross-linguistic near-synonyms are a reality. And there is no fundamental distinction between cross-linguistic near-synonyms and near-synonyms within a single language; they could all be grouped into one huge usage note. Indeed, there are books for second language learners that explicitly group and differentiate cross-linguistic near-synonyms (Batchelor and Offord, 1993; Farrell, 1977). For example, figure 3.3 gives Farrell's usage note for the German near-synonyms of *mistake*. Thus, we claim that all of the types of difference classified here should apply equally well across languages.

It is especially evident how cross-linguistic sets of near-synonyms can overlap in denotation and extension. In the simple cases, two words coincide precisely: *fork* (Engl.) : *fourchette* (Fr.). Sometimes, two words in one language will cover a single word in another. For instance, as figure 3.4 shows, *river* corresponds to two French words, *rivière* and *fleuve*, that are distinguished in terms of size and whether the watercourse flows into another watercourse or an ocean. Actually, the diagram glosses over some of the details: while *rivière* can apply to any significant watercourse, *fleuve* is for a larger watercourse with many tributaries that flows into an ocean (Bénac, 1956). There are also several other near-synonyms including *brook, creek,* and *rill* in English, and *ru* and *torrent* in French.

In more complex cases, the near-synonyms of separate languages each cover the space differently. For instance, figure 3.5 shows how some of the words for tracts of trees differ across five languages. (And, even within the same language the words have a fuzzy overlap in extension, as depicted by the jagged lines.) While the distinctions between the English words and the French words are the same, the distinction between the German words and the English words breaks at a different point. Dutch has three words that cover the same space, and Danish just one. So, for example, *woods* can be translated into German as either *Wald* or *Gehölz*, depending on the circumstances.

In the most complex cases, it is difficult to even draw a diagram of the overlaps, because the near-synonyms significantly overlap with respect to extension within each language. Consider the many near-synonyms of *error* in French, English, and German shown in figure 3.6. The links

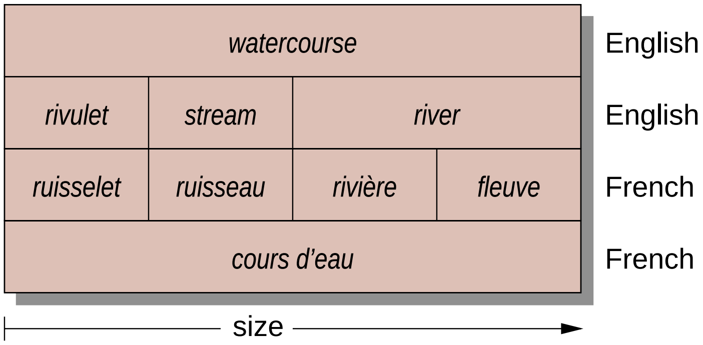


Figure 3.4: Words for watercourses in English and French.

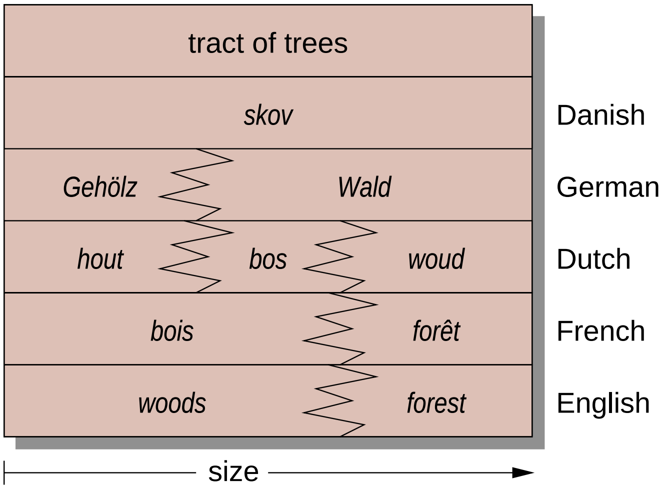


Figure 3.5: Words for tracts of trees in Danish, German, Dutch, French, and English. The data was consolidated by DiMarco, Hirst and Stede (1993) from other sources. Jagged lines were added to show the fuzzy overlap.

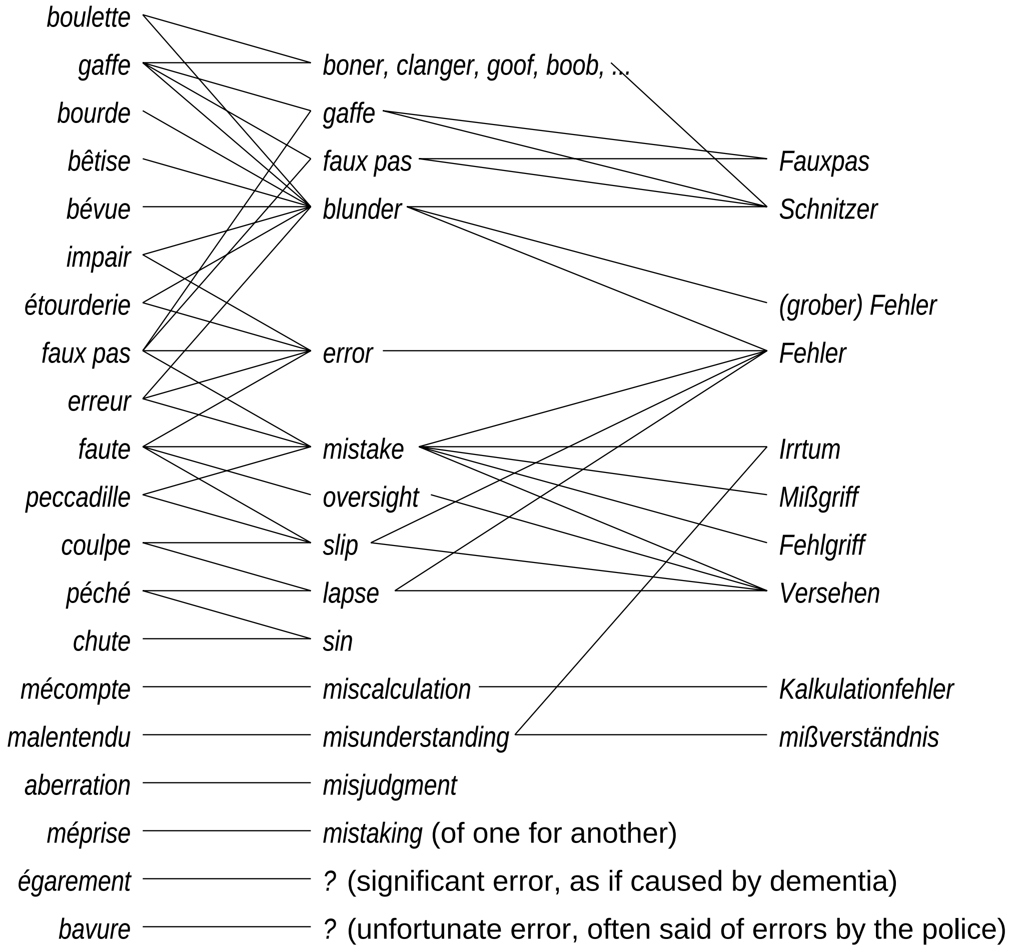


Figure 3.6: Correspondences between the French, English, and German near-synonyms of *error*.

were assigned according to Batchelor and Offord's (1993) French–English entry, Bénac's (1956) usage notes for *erreur* and *faute*, Farrell's (1977) English–German entry for *mistake*, and *Langenscheidt's New College German Dictionary* (Langenscheidt, 1995). Links indicate similarity of meaning (or extension), rather than difference in meaning. French appears to have more words for *error* than English, and far more than German. Notice that French has a few specialized words for which English and German have no equivalents. That is, English and German have *lexical gaps* for *bavure* and *égarement*. Also, while *méprise* has an equivalent verb form in English (*to mistake*), there is no appropriate noun. Lexical gaps are the converse of overlaps, and are quite frequent between languages.

3.7 Multi-dimensional variation

As we mentioned earlier, near-synonyms can vary on many dimensions at the same time. As an example, consider the *error* cluster. By analyzing Gove's usage note for this cluster (see figure 1.1, page 3), we came up with the following dimensions, among others, on which the near-synonyms of *error* vary simultaneously:

- **Stupidity:** A binary denotational dimension, it applies to the person who commits the error.
- **Blameworthiness:** A continuous denotational dimension, also applies to the person who commits the error.
- **Severity of criticism:** A continuous denotational dimension of which the criticism applies to the actor of the error. The criticizer is not necessarily specified.
- **Misconception as cause:** A binary denotational dimension referring to the cause of the error, it has as actor the person who commits the error.
- **Accident as cause:** A binary denotational dimension similar to the previous dimension.
- **Attitude:** An expressed attitude that is directed towards the person who commits the error.
- **Formality:** A stylistic dimension. (*Howler* and the other colloquial terms are informal, but we have no data on the rest of the near-synonyms.)
- **Concreteness:** A stylistic dimension. (*Blunder* is concrete, and *error* and *mistake* are more abstract (DiMarco, Hirst and Stede, 1993)).

Clearly, we have glossed over many of the distinctions, and have left out others. However, we do not claim that these form the best set of differentiae for this particular cluster, only that they are representative of the variety of differentia that we would like our model to account for.

3.8 Summary

In this chapter we classified many of the ways in which near-synonyms can differ; table 3.14 summarizes our classification. And from the proceeding discussion of near-synonymic variation, it should now be clear that for a model (of lexical knowledge) to properly account for near-synonymy, it it will have to incorporate solutions to the problems related to at least the following phenomena:

- The four main types of variation (denotational, stylistic, expressive, and collocational) are qualitatively different, so each must be handled in its own way.
  - Denotational variation involves differences in the infinite range of concepts or ideas that words can express. As such, a particular dimension of denotational variation might be relevant and meaningful only to a small set words. (E.g., *error* and *mistake* express different levels of severity of criticism.)
  - Stylistic variation involves differences in a small finite set of dimensions on which all words can be compared. (E.g., *pissed* is informal, as is *fib*, but *inebriated* and *lie* are more formal.)
  - Expressive variation involves variation on emotive dimensions such as ‘intimacy’ (e.g., *daddy* : *father*), and in expressed attitude (e.g., *slim* : *skinny*). Expressive variation is similar to stylistic variation, but unlike a stylistic distinction, a distinction in

expressed attitude explicitly involves a participant of the situation being expressed (e.g., the person that is *slim* or *skinny*), and also the speaker who is expressing the attitude.

  - Finally, collocational variation involves the words or concepts with which a word can be combined, possibly idiomatically, in a well-formed sentence. A representation of collocational variation will have to allow references to lexical items and concepts. (E.g., *task* must point to *daunting* as a preferred modifier, and vice versa.)

- The manner in which nuances of meaning are expressed can vary; they can be conveyed indirectly, and with varying strength. There appears to be a continuum of indirectness from 'suggestion' to 'implication' to 'denotation'. Many seemingly denotational nuances cannot be represented by simple necessary and sufficient truth-conditions. (E.g., the 'lower severity of criticism' expressed by *mistake* is merely implied, not denoted.)

- Meanings, hence differences, can be fuzzy. (E.g., *forest* and *woods* have a fuzzy difference in terms of size, wildness, and so on.)

- Differences can be multi-dimensional, and the dimensions can be interdependent. (E.g., *blunder* and *mistake* differ on several denotational dimensions, as well as on stylistic and attitudinal dimensions.)

- Differences are not just between simple features, but involve concepts that relate roles and aspects of a situation. (E.g., 'severity of criticism' is a complex concept that involves both a 'criticizer' and a 'criticizee'.)

- Context influences the relevance of lexical differences.

Variation category	Variation type	Example
Stylistic	Geographical dialect	*loch* : *lake*
	Temporal dialect	*lapidate* : *stone (to death)*
	Social dialect	*loo* : *toilet*
	Language	*verboten* : *forbidden*
	Sublanguage	*matrimony* : *wedlock* : *marriage*
	Formality	*pissed* : *drunk* : *inebriated*
	Force	*ruin* : *wreck* : *destroy* : *annihilate*
	Concreteness	*name* : *christen*
	Floridity	*house* : *habitation*
	Euphemism	*toilet* : *bathroom* : *washroom* : *restroom*
	Familiarity	*divulge* : *disclose* : *reveal* : *tell*
	Simplicity	*hound* : *dog*
Expressive	Emotive	*daddy* : *dad* : *father*
	Expressed attitude	*skinny* : *thin* : *slim*
Denotational	Denotation	*account* : *chronicle* : *report*
	Implication	*mistake* : *slip* : *lapse*
	Suggestion	*help* : *aid* : *assist*
	Emphasis (rel. to concept)	*scurrility* : *abuse*
	Emphasis (rel. to word)	*enemy* : *foe*
	Frequency of expression	*version* : *story*
	Fine-grained technical	*alligator* : *crocodile*
	Abstract dimension	*seep* : *drip*
	Continuous dimension	*mistake* : *error* : *blunder*
	Binary dimension	*escort* : *accompany*
	Multi-value dimension	*order* : *command*
	Complex ‘dimension’	*begin* : *start* : *initiate*
	Specificity	*eat* : *consume* : *devour* : *dine* : *gobble*
	Extensional overlap	*high* : *tall*
	Fuzzy overlap	*forest* : *woods*
Collocational	Selectional restrictions	*land* : *perch*
	Lexical collocation	*task* : *job*
	Idiom	*bite the dust* : ∗*gnaw the powder*
	Grammatical collocation	*correct* : *right*
	Subcategorization	*teach* : *instruct*
	Converse	*buy* : *sell*

Table 3.14: Classification of lexical variation with examples.

Chapter 4

Near-synonymy in lexical semantics and lexical choice: A survey

> 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*.

In this chapter I will show that near-synonymy is not adequately accounted for in current theories of lexical semantics or in computational systems. In fact, most current models and systems adopt the kind of rigid representation suggested by Big Brother's dictum in Orwell's *Nineteen Eighty-Four*. In doing so, these models and systems each fail to address most or all of the problems of representing near-synonyms that we listed in section 3.8.

Note that there is no need for a comprehensive survey of lexical semantics or of lexical choice since these subjects have been competently covered elsewhere: see Cruse (1986), Lehrer and Kittay (1992b), or Lyons (1995) on the former, and Robin (1990), Saint-Dizier and Viegas (1995), or Stede (1995) on the latter.

4.1 Lexical semantics

If semantics is the of study of meaning, then *lexical semantics* is the study of the word meaning, or of how meaning can be encoded in words, or *lexicalized*. In order to study near-synonyms, we must take a fairly broad perspective on meaning that includes not only *propositional* meaning but also *expressive* meaning, including aspects of emotion, attitude, and style (Cruse, 1986; Lyons, 1995). In this section I will discuss the following (not necessarily mutually exclusive) general theories of meaning, which are relevant to our goal or representing near-synonyms:

Decompositional theory The meaning of a word is decomposed into a set of primitive or elementary components.

Structural theory The meaning of a word depends on its relationships with other words.

Contextual theory Meaning is determined by, or is equivalent to, a word's use in actual language.

Prototype theory A word's meaning is represented by an exemplar (a best example) or some abstraction thereof.

While each type of theory accounts for many aspects of lexical meaning, none makes explicit provisions for near-synonymy and its ensuing problems. Indeed, many simply ignore it or treat it very simplistically. Nevertheless, we can still investigate how the theories would handle near-synonymy if pushed to do so. We examine each of the theories below.

4.1.1 From lexical decomposition to ontologies

Most systems that do lexical choice, as will be discussed in section 4.2, use some form of lexical decomposition to represent meaning. The most sophisticated decompose word meaning into concepts represented in an ontology. This section discusses the theory as it relates to near-synonymy.

In *lexical decomposition*, or componential analysis, the meaning of a word is decomposed into semantic primitives. The theory has a long history, but was developed, most notably, by Katz and Fodor (1963) and later by Wierzbicka (1972), Schank (1975), Jackendoff (1983; 1990), Goddard (1994), and others. The three main aspects of the theory are that (1) a word's meaning can be decomposed into a set (or structure) of more general components, or semantic features, (2) the components are primitive (i.e., undecomposable) and universal, and (3), though not always stated, the set of components is exhaustive, that is, the components are individually necessary and collectively sufficient to define the word. For example, *bachelor* can be decomposed into the set {ADULT, MALE, NOT-MARRIED}, and *kill* into the structure (CAUSE (X, DIE (Y))). While intuitively plausible—after all dictionaries assume the theory to some extent—and practical computationally, for it provides a systematic way to represent meaning across languages, assuming the universality of the features,[1] the theory has been subjected to many criticisms, not the least of which is that it can only account for rather simple examples (Lyons, 1977). In all of its various interpretations it cannot adequately account for near-synonyms.

In the theory, different near-synonyms would be represented by almost identical sets of features. So for instance, the near-synonyms of *error* might share a whole set of features, whatever they are, but differ in the following way: only *error* might have the features BLAMEWORTHY and SEVERE, only *mistake* might have MISCONCEPTION and NON-BLAMEWORTHY, only *blunder* might have STUPID, and so on. But such a representation is inadequate for the following reasons:

1. It is too simplistic; it does not account for the fact that features ought to represent relations between aspects and participants of a situation to which a word refers. For instance, MISCONCEPTION is a relation between the agent of the mistake and the cause of the mistake.

2. It does not account for relations between the features themselves; for instance, the misconception involved in a mistake may be related to its non-blameworthiness.

3. It does not account for the indirect expression of meaning, because the set of features that define a word are considered to be necessary and sufficient.

4. It does not allow one to represent fuzzy distinctions (e.g., the difference in severity between *error* and *mistake* is fuzzy).

It is important to resolve these four points because they are crucial to the process that maps from input to words.

[1] The criticism that primitive and universal features may be impossible to discover is not relevant here, as we are more concerned with whether a feature-based model is adequate at all for representing synonyms.

$$\textit{error:}\ \left[\begin{array}{ll}\textsc{concept} & \text{GenericError}\\ \textsc{agent} & \left[\begin{array}{ll}\textsc{concept} & \text{Person}\\ \textsc{attribute} & \left[\textsc{concept}\ \text{Blameworthy}\right]\end{array}\right]\\ \textsc{attribute} & \left[\textsc{concept}\ \text{Severe}\right]\end{array}\right]$$

$$\textit{mistake:}\ \left[\begin{array}{ll}\textsc{concept} & \text{GenericError}\\ \textsc{agent}\ \boxed{1} & \left[\begin{array}{ll}\textsc{concept} & \text{Person}\\ \textsc{attribute} & \left[\textsc{concept}\ \text{NotBlameworthy}\right]\end{array}\right]\\ \textsc{cause} & \left[\begin{array}{ll}\textsc{concept} & \text{Misconception}\\ \textsc{agent}\ \boxed{1} & \end{array}\right]\end{array}\right]$$

Figure 4.1: Possible denotations of *error* and of *mistake* in a frame semantics.

One extension to the simple feature-based model, which resolves the first point above, is to use structures of semantic primitives. For example, Jackendoff's (1983; 1990) rather influential representational system of *lexical conceptual structures* (LCSs) represents meaning in terms of structures of semantic primitives, but what makes the system different is the principle used as a basis for specifying the primitives. A primitive is adduced only if it appears responsible for a specific syntactic and / or (linguistic) semantic behaviour. This restricts the system to accounting for only syntactic / structural phenomena. For example, in Dorr's (1993; 1994) use of LCSs for machine translation, she explicitly does not account for pure lexical divergences between languages as these do not have a reflection in syntax (e.g., her system does not account for how *river* can be translated as either *rivière* or *fleuve*, because the difference is not structural.) Such divergences would be represented by simple features in the system. (Compare work on lexis in systemic functional linguistics in section 4.2.2 below.)

We can also extend the simple-feature account to use a higher-level (conceptual) semantics by considering every feature to be a concept in a broad sense. If we formally define features as concepts in a suitable system, then not only can complex properties be represented, but some forms of reasoning about the properties becomes possible. One could represent concepts by nodes in a semantic network (e.g., Sowa's (1984) conceptual graphs) or, more generally, by frames containing attribute–value pairs that represent properties and relations (to other concepts) (Fillmore, 1982; Barsalou, 1992). A word would be linked to (or decomposed into) a frame that specifies its denotation. For example, figure 4.1 shows possible denotations for *error* and *mistake* in a *frame semantics*. Each frame is a list of attributes (in uppercase) and their associated values, which are other frames or references to other frames (capitalized terms). Thus, a frame represents a concept, and an attribute a relation between concepts. The other synonyms of *error* would be represented in similar frames with possibly different values for the same attributes or even different attributes. Brachman and Schmolze (1985) formalized frame semantics—actually a subset of it called description logic—in the KL-ONE knowledge representation system, which is essentially a formalization of lexical decomposition. KL-ONE has spawned several specific implementations including Loom (MacGregor and Bates, 1987).

But as with LCSs, we require a principle to decide which concepts to represent. The basic

Level	Example
Class	14: THE MIND AND IDEAS
Subclass	972 TRUTH, 973 WISE SAYING, . . . , 976 DISILLUSIONMENT
Category	974 ERROR
Paragraph	974.3 mistake, error, *erratum* <L>, *corrigendum* <L>; fault <Fr>; gross error, bevue; misconception, misapprehension, . . .
Cluster	mistake, error, *erratum* <L>, *corrigendum* <L>
Word	mistake

Table 4.1: The system of classification of *Roget's International Thesaurus* with examples (Chapman, 1992).

principle followed in current MT and NLG systems is to define concepts that correspond to things in the real world (or within a limited application domain). They are intended to be universal, though not necessarily primitive, in the sense that they are language-independent or, at least, language-neutral. This is a theoretical requirement for interlingual machine translation, multilingual text generation, and reasoning in general. The set of concepts and their interrelationships thus defined form an *ontology*. An ontology is a particular model of the world, or, a model of the existence of things in the world and their relationships. See, for example, work on PANGLOSS / SENSUS (Nirenburg, 1995; Knight and Luk, 1994), CYC (Lenat, 1995), Mikrokosmos (Mahesh and Nirenburg, 1995; Nirenburg, Raskin and Onyshkevych, 1995), WordNet (Miller et al., 1990; Fellbaum, 1998), EuroWordNet (Vossen, 1998),[2] and work on constructing such knowledge bases automatically (Richardson, 1997; Barrière, 1997). Of course, the idea of using an ontology for organizing a lexicon is not new: Roget, in 1805, had conceived of his Thesaurus as a classification of words according to ideas, and while the classification scheme has been modernized over the years, his original idea of using a hierarchical structure remains (see table 4.1).

Of course, coming up with the 'right' ontology is a very controversial issue, but we can assume that an appropriate ontology will in due course be constructed. In fact, the ANSI Ad Hoc Group on Ontology Standards made the first step in November 1997 in adopting a draft reference ontology. It consists of about 3000 upper-level concepts merged together from CYC, PANGLOSS, WordNet, and Penman. Unfortunately, development of this particular endeavour has recently stalled, so its future is uncertain. (We will continue this line of discussion in section 7.2.)

Now, it is natural to organize the ontology into a hierarchy, or taxonomy, from the most general concepts down to the most specific, and to use inheritance. This model is now conventional for lexical representation: the denotation of a lexical item is represented as a concept, or a structure of concepts (i.e., a word sense is linked to the concept that it lexicalizes). See for instance the various papers in Evens (1988) and Pustejovsky and Bergler (1992). (Actually, this model

[2] According to our definition of 'ontology', WordNet and EuroWordNet are not real ontologies, since they are concerned more with lexical organization than conceptual organization. That is, they do not contain concepts per se, but lexical items.

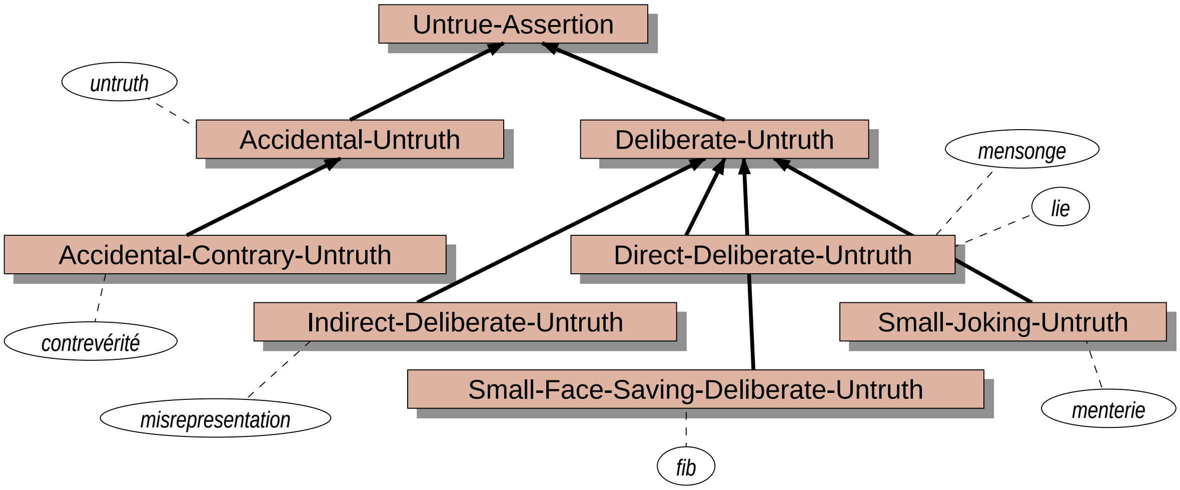


Figure 4.2: Hirst's (1995) proposed hierarchy for the various English words for untrue assertions (i.e., lies) augmented with French words.

shades into the relational model of the lexicon to be discussed in the next section.) So except for polysemy and absolute synonymy, there is no logical difference between a lexical item and a concept. Therefore, words that are nearly synonymous would have to be linked each to their own slightly different concepts. The problem comes in trying to represent these slightly different concepts and the relationships between them. Hirst (1995) shows that one ends up with an awkward proliferation of language-specific concepts, contrary to the interlingual function of the ontology.[3] For instance, figure 4.2 shows Hirst's hierarchy of concepts for the group of *lie* 'untrue-assertion' synonyms (based on Gove (1973)) augmented with a few French synonyms (based on Bailly (1947)). Even in this relatively simple case, notice how one must define some concepts that are specific to only English, and others that are specific to only French. And every other language that the system has to deal with would probably require its own specific concepts. It is difficult to develop an efficient lexical choice process for machine translation using such a hierarchy. While some words have obvious translations, such as *mensonge* to *lie*, how would it translate *menterie*? *Fib* seems closest in meaning, but, in this hierarchy, it is linked to a different concept.

Of course, it might be better to represent such word meanings in a lattice rather than in a strict hierarchy as shown here. In the same manner as in a frame semantics, each concept in the lattice would connect to many other concepts that represent the various symbolic features, such as 'deliberateness', that are associated with concepts of telling a lie (in this example). But even if we are not concerned with language-independence, and even if we disregard the complexity and engineering disadvantages of such as approach (a subject that we will discuss further in sections 4.2.4 and 5.1 below), it is still inadequate for representing aspects of word meaning that

[3] The proliferation of language-specific concepts required under the conventional model is not unlike the problem of the proliferation of senses seen in polysemy (Cruse, 1995; Pustejovsky, 1995; Rais-Ghasem, 1998). For instance, the sense of *begin* in *to begin a book* could be 'begin reading', 'begin writing', 'begin editing', etc. The predominant solution to account for polysemy is to use a generative model that 'generates' from a more abstract form a specific sense as needed in the context. Perhaps a similar solution that generates 'word meanings' as needed from more general representations is necessary for near-synonymy as well.

are not purely symbolic and denotational.

Such a model cannot represent the fine-grained indirect and implicit meanings that many near-synonyms differ in conveying. That is, it cannot represent aspects of word meaning that do not contribute to the truth-conditions or logical entailments of a sentence (Frege, 1892; Tarski, 1944). (Certainly, the model cannot handle other non-propositional aspects of meaning such as style, emotion, and attitude.) Indeed, it is now accepted that most words cannot be defined solely in terms of necessary and sufficient features. Lehrer (1974), for instance, suggests that both *defining* and *optional* features might be necessary. And Cruse (1986) suggests the following range of types of feature differing in their degree of necessity: *criterial* (i.e., essential), *expected*, *possible*, *unexpected*, *excluded*, and further divides expected features into *canonical* and *non-canonical* features. The argument has even been made that no feature is essential in the definition of a word in every context (Wittgenstein, 1953) (see section 4.1.4 below). Even Katz and Fodor had to distinguish between *markers*, which represent the systematic components of meaning, and *distinguishers* for the idiosyncratic residue. Lyons adds that lexical decomposition "runs into quite serious problems, if it is pushed beyond the prototypical, or focal, meaning of expressions" (1995, p. 217).

Another problem is that this approach does not resolve point 4 above. Features and concepts are an all-or-nothing affair, so representing words that have fuzzy meanings such as *forest* with the features like LARGE and WILD is inadequate—even LARGE and WILD are fuzzy concepts.

Despite these criticisms, we shouldn't abandon the idea of using lexical decomposition or ontological concepts in our representations—it is after all the dominant means of lexical representation in computational systems. Some distinctions are probably best expressed as features; for instance, *seep* and *drip* differ in terms of the feature INTERMITTENT (DiMarco, Hirst and Stede, 1993). And while lexical decomposition seems inadequate for representing the fine-grained aspects of meaning, it can satisfactorily account for the more coarse-grained aspects of meaning, as demonstrated by Jackendoff's LCSs. Furthermore, the ontology usually provides powerful inference mechanisms that can be used for reasoning in many tasks of natural language processing. It facilitates the efficient construction of meaning representations from text, and, in text generation, the efficient choice of words (at least at a coarse grain) and syntactic structures (see section 4.2.5 below).

4.1.2 Structuralism

Structuralism, the theory that all the units of a language derive their meaning solely from their relationships with other units, has its modern origins with Saussure (1916, p. 114), who says:

> In a given language, all the words which express neighbouring ideas help define one another's meaning. Each of a set of synonyms like *redouter* ('to dread'), *craindre* ('to fear'), *avoir peur* ('to be afraid') has its particular value only because they stand in contrast with one another. ... No word has a value that can be identified independently of what else is in its vicinity.

So, in this theory, words—one kind of linguistic unit—are but points in a network of relations of contrast and equivalence. And as such they have no internal structure and are not decomposable into components of meaning. Structuralism has developed in two separate lines of research: the theory of lexical fields (Trier, 1934; Lehrer, 1974), and relational theories of word meaning (Lyons, 1977; Cruse, 1986; Evens, 1988).

As discussed above, a decompositional theory of meaning can shade into a structuralist theory when the components or primitives are themselves related to one another. Conversely, a structuralist theory can be developed into a decompositional theory when one seeks to represent how related items demarcate each other. Lexical-field theory has taken the latter direction. A lexical field is a group of lexical items that totally cover a 'cohesive' semantic field, and individually demarcate positions within the field (as a mosaic, in Trier's original formulation, but overlaps and gaps are allowed in more current versions (Lehrer, 1974)). A set of near-synonyms can be considered a very restricted lexical field. Coseriu (1967) proposed two types of semantic component to distinguish elements within a field: *semes* and *classemes*. The former are primitive components of meaning that operate within a single field, and, crucially, serve to structure the field in terms of *opposition*, while the latter are more general and cut across many different fields.

Relational theories have been more prominent in computational linguistics. The four most widely accepted lexical relations[4] are *synonymy*, *antonymy*, *hyponymy*, and *meronymy* (part-whole), but many more have been suggested (for instance, Mel'čuk and Zholkovsky (1988) define about 60 relations, though some are syntagmatic). What one finds is that it is very difficult to restrict relations to being lexical (between words) rather than semantic (between concepts) since words and concepts are so interrelated themselves. Hyponymy, for example, might better be thought of as a relation between concepts for it is not really the word *car* that is a hyponym of *vehicle*, but the concept 'car' (or the denotation of *car*) that is a kind of 'vehicle'. This applies even more so to meronymy. Thus, most models use a combination of lexical relations and semantic relations focusing on one or the other depending on the purpose. Many even conflate (or confound) the two as *lexical-semantic relations* or as *sense relations*. For instance, in WordNet (Miller et al., 1990) synonymy is a lexical relation that groups words into *synsets*, which can be thought of as concepts. All of the other relations are semantic as they relate synsets, but they can also be thought of as lexical because they indirectly relate the words in the synsets.

Current relational models of the lexicon run into problems in representing near-synonyms precisely. In WordNet, for instance, synonyms are considered absolute and grouped into synsets. EuroWordNet includes a near-synonymy relation for relating synsets that are not directly equivalent across languages (Vossen, Díez-Orzas and Peters, 1997; Vossen, 1998). Mel'čuk and Zholkovsky (1988) define one relation for absolute synonymy and three more for narrower, broader, and "intersecting" (quasi-) synonyms. (Mel'čuk and Zholkovsky give the examples that *shoot* 'discharge a firearm' is a synonym for *fire*, and that *shoot* is a narrower synonym for *fire upon*.) Cruse and Lyons, as mentioned in section 2.2, define several types of relation as well, but find that the relations are inadequate since synonymy is a matter of degree on many dimensions that cannot be represented by a standard relation. Cruse even suggests that some groups of near-synonyms simply lack any kind of structure. The problem is that standard relations of synonymy can only tell us about degree of similarity—they do not elucidate the differences.

Actually, current relational models have moved so far from Saussure's structuralism that they have forgotten that opposition and contrast are fundamental to his theory. For Saussure, "in a language there are only differences, *and no positive terms*" (p. 118). Clark's (1992) Principle of Contrast also implies from a psycholinguistic perspective that near-synonyms must be

[4] Two general types of relation are usually distinguished. Paradigmatic relations relate words that can be substituted for one another in the same sentence, and so are the more common sort of lexical relation discussed in the literature on relational theories. Syntagmatic relations relate words that co-occur in expressions. They take in phenomena ranging from traditional syntax to collocations and idioms, and so will be covered in the next section on contextual theories of meaning.

related by difference. Hirst (1995) promotes this point and argues that differences should be represented as objects that can be reasoned about and compared with one another. However, very little research has been done on how one should do this. DiMarco, Hirst and Stede (1993) adduced 26 dimensions along which words can vary (for examples, see table 3.7, page 34), and DiMarco (1994) shows how such dimensions can be composed to form explicit relations of difference. But as discussed in the previous section, simple dimensions (which are equivalent to features) are inadequate to represent differences between near-synonyms in general, so composing them would not solve the problem. Moreover, it is difficult to account for differences in the conventional (taxonomic) model of the lexicon, which represents meaning in only positive terms.

However, it is not a given that near-synonyms necessarily contrast. Taylor (1997), for one, rejects the Saussurean notion of contrast, but doesn't deny there may be implicit differences in meaning. (Note also that Cruse (1986) states that to be near-synonyms words must exhibit a low degree of contrastiveness, and indeed his linguistic tests for contrast fail to work for many near-synonyms such as *tall* and *high*.) Taylor claims, instead, that many near-synonyms may be in a relation of *coextension* in that they cover an overlapping area (or even the same area) of meaning, but place the focus differently. Thus, each near-synonym puts a different perspective on the same meaning. However, in its current stage of development, the theory would not be easy to implement.

4.1.3 Context: Meaning is use

The two central aspects of lexical meaning, according to Firth (1957), are the "context of situation" and the syntagmatic pattern of *collocation*. Both involve context, broadly construed. The former can be called global context as it involves the environment in relation to which a text is produced or understood including interpersonal relationships and the domain of the discourse. The latter, which we call local context, involves the immediate sentence or expression in which a word is used.

Global context is not very well understood, at least not enough to be formalized in a computational system. The most comprehensive theory is the theory of register within systemic functional linguistics (Halliday, 1978; Martin, 1992). The *register* of a text consists of three variables: field, mode, and tenor, which refer to the subject matter, the means of communication, and the social relationships among the participants, respectively. The theory has been applied mostly to the descriptive analysis of texts (by hand), so it is not yet clear how these high-level representations map to word choice. Related to this point, in a large empirical study, Biber (1988) was able to find correlations in the occurrence of 67 linguistic features across different genres of text and so identify six general textual dimensions of (1) involved versus informational production, (2) narrative versus non-narrative concerns, (3) explicit versus situation-dependent reference, (4) overt expression of persuasion, (5) abstract versus non-abstract information, and (6) on-line informational elaboration. However, Biber's results are also mainly descriptive.

Simplifications have been attempted for NLG systems, but they sacrifice a principled theory. For example, Bateman and Paris's (1991) model associates each register with a set of seemingly ad hoc lexicogrammatical features that force the generator to make certain choices. With a similar intent, Stede (1993) represents each genre as a numeric 'profile' of several stylistic dimensions. Words are then chosen that match a given profile as closely as possible.

Local context, on the other hand, has been given more attention, especially because statistical techniques have proved so useful in applying it. In essence, lexical items are in a relation of mutual expectancy, so the use of a word may be predicted by others in its local context

(Firth, 1957). Sinclair (1966) sought to validate and extend these theoretical claims by the statistical and empirical study of word co-occurrence in large corpora. One of his goals, in possibly the first application of local context, was to create a large dictionary of English collocations, but as he lacked powerful computers, his efforts came too soon. Since then, with the advent of these computers, dictionaries of collocations have appeared (e.g., Benson, Benson and Ilson (1986)), the ideas have been formalized (Hausmann, 1985; Mel'čuk and Polguère, 1987), and methods of automatically acquiring collocations from corpora have been devised (Smadja, 1991; Church et al., 1991). Collocations have also been applied in NLG and MT systems (Iordanskaja, Kittredge and Polguère, 1991; Smadja and McKeown, 1991; Smadja, McKeown and Hatzivassiloglou, 1996), but not specifically for modelling near-synonyms.

Local context can be formalized in an evidence-based model that represents context as a set of features, including words, morphology, collocations, syntactic structure, and semantic classes, that are observed to co-occur with, and thereby predict, a word or word sense. Such models have been applied most successfully to word sense disambiguation, but also to spelling correction (Yarowsky, 1992; Grefenstette, 1994; Ng and Lee, 1996; Golding and Schabes, 1996; Karov and Edelman, 1998; Schütze, 1998). But with near-synonyms and lexical choice, little work has been done. Church et al. (1994) show that two near-synonyms can be differentiated in terms of their difference in co-occurrence to other words; however this method is inefficient in time and space. Every pair of words in a language must be compared to every other word in the language, so the tables built for each word pair are the size of the lexicon. Edmonds (1997) shows how to overcome the space-inefficiency by using networks of lexical co-occurrence relations, and also shows how to apply the model to lexical choice. This work will be reported in greater detail in chapter 6.

4.1.4 Prototype theory

Prototype theory as an alternative to the classical truth-conditional and decompositional theories of meaning representation was firmly established by Rosch (1978), but it was Wittgenstein (1953) who first argued against the classical view. Wittgenstein explained that the category of 'games' has no common properties (i.e., necessary and sufficient features) and that its members are related instead by family resemblance. Rosch (1978) extended and generalized this idea on the basis of evidence from her many empirical experiments that show the pervasiveness of *prototype effects* exhibited by categories (including concepts denoted by lexical items). The basic prototype effect is that some members of a category are deemed more central or typical members of a category than others. Some categories are graded, like 'tall man', with fuzzy boundaries and degrees of membership for its members. Others have clear boundaries, like 'bird', but still have some members that are better examples of the category than others. Such members are variously called *prototypes* or *exemplars*, depending on the theory. The classical view cannot account for prototype effects because it implies that all members of a category are equal—none is more of a member than any other. Prototype theory explicitly allows concepts to be fuzzy, and features, whatever they may be, to not be necessary.

Prototype effects and categorization in language have been taken up in a field called *cognitive linguistics* (Taylor, 1989). And although near-synonyms have not been specifically addressed in the field, one surmises that prototype theory might be appropriate and effective for modelling near-synonyms. Every near-synonym could be said to represent a category of possible referents or of uses, which has a prototype or exemplar. This would account for the fuzzy distinctions between some near-synonyms, and for nuances of meaning that cannot be modelled by necessary conditions. But does each near-synonym (of a set) represent a separate category, or does each have a different prototype in the *same* category? Hirst (1995) criticizes Lakoff (1987) for apparently taking the former position, because the position requires a simplistic view of fine-grained lexical meaning that is not psychologically plausible. Along with Hirst, Taylor (1989) and Geeraerts, Grondelaers and Bakema (1994) also favour the latter view. Taylor's view is a result of his theory of coextension (see above), and Geeraerts, Grondelaers and Bakema use the view in their account of lexical variation in context. In their account, lexical choice is done according to three criteria: (1) choose a category to which the entity (to refer to) typically belongs; (2) choose a word with the highest salience of the set of alternatives (possibly the one whose prototype is closest to the input, however that is computed); and (3) allow contextual factors to affect (1) and (2).

The problem with prototype theory is that it is very difficult to make computationally precise. Take Lakoff's (1987) *idealized cognitive models* (ICMs), in one of the more extensive versions of the theory. In the theory, knowledge is organized by means of ICMs, which are gestalts structured as schemas, frames, or according to a few other principles—prototype effects are but byproducts of this organization. An ICM sets up the ideal background conditions in which to understand (or lexicalize) a concept. This much can be formalized, but for ICMs to be useful one must be able to compare them, to determine similarity, for instance, to each other and to other cognitive models of situations in the world, and this is very difficult to formalize.

The few attempts to formalize prototype theory in computational systems have inevitably oversimplified the theory (see Lakoff's (1987) discussion). For instance, Smith and Medin (1981) analyze many methods of moving beyond features as necessary-and-sufficient, one of which is to extend the classical view so that features are assigned probabilities of being defining features of a concept. Similarly, Dubois and Prade (1988) use Zadeh's *fuzzy set theory* (1965)—a theory for representing graded concepts—to implement their possibility theory, which they apply to fuzzy relational databases. For instance, if a record represents the properties of a person, then age might have the value 'about 20' rather than exactly 20. So values of record attributes can be fuzzy.

In summary, after further research, prototype theory should be able to solve some of the problems in representing near-synonyms that we listed in section 3.8 including the representation of fuzzy, multi-dimensional, and contextual differences between near-synonyms. However, much more basic research is required before it can be formalized sufficiently for a computational system, and even then, it is unlikely to solve all of the problems.

4.2 Lexical choice

This section reviews the research to date on the treatment of near-synonymy in lexical choice in MT and NLG systems. While no system has explicitly dealt with near-synonymy, many can handle synonyms and near-synonyms, though often very simplistically as absolute synonyms. To begin with I will briefly review the basic methods of MT and NLG.

There are two basic approaches to MT: interlingual methods and transfer methods. The former assumes a language-neutral formalism for representing meaning unambiguously, called an *interlingua*, that is used as a 'stopover' point in any translation from a source language to a target language. The level, or depth, of knowledge representation required of an interlingua depends on the needs and purpose of each particular system; for instance, in knowledge-based MT (KBMT) (Nirenburg et al., 1992) and PANGLOSS (Nirenburg, 1995), pragmatic, conceptual, and semantic knowledge is represented, but in UNITRAN (Dorr, 1993) only a 'linguistic' semantic level is. Lexical choice occurs during the mapping from an interlingual representation to surface representation. The transfer approach, on the other hand, uses no intermediate formalism; the source text is translated directly to the target language according to a model built specifically for individual pairs of languages. Lexical choice occurs by applying transfer rules to expressions in the source text. The statistical approach to MT (Brown et al., 1990) is a transfer approach, as is 'shake-and-bake' translation (Whitelock, 1994). Most commercial MT systems today use a transfer approach, because the approach allows for a broad and robust, but shallow, coverage of linguistic phenomena. Finally, there are also hybrid approaches such as JAPANGLOSS (Knight et al., 1995).

Generally, an NLG system takes some form of 'semantic' or non-linguistic information as input, and outputs a sentence, paragraph, or larger surface form. Thus, NLG can be considered the second half of a pure interlingual MT process, as indeed is made explicit in some MT systems (e.g., in DIOGENES (Nirenburg and Nirenburg, 1988) and in the Mikrokosmos project (Beale et al., 1998)). Thus, a typical NLG system takes input in the form of concepts, at whatever level of description is necessary, or instances of concepts. The primary purpose of lexical choice is to find the words that lexicalize the concepts, though in practice lexical choice must involve many more tasks, because it must lexicalize concepts in such a way that they can be assembled into a well-formed surface form (both pragmatically and syntactically).

4.2.1 One-to-one concept–word links

Many early NLG systems hardly *chose* words at all because concepts and words were related one-to-one in the systems. For example, MUMBLE (McDonald, 1983; McDonald, 1987) takes input in the form of semantic propositions. Objects in the propositions (MUMBLE's concepts) are linked directly to words, so lexical choice is rudimentary. For instance a semantic object such as `door-3` is simply replaced by the word *door*. McKeown's TEXT (McKeown, 1985) is slightly more versatile because semantic objects map to different words depending on syntactic constraints. For example, a semantic instance of `guided`, an object representing a guided missile, would select *guided* when an adjective is required, and *projectile* when a noun is required. It is not necessary to have a separate concept for the property of being 'guided'.

Since these systems use a one-to-one link between concepts and words, one could represent near-synonyms by creating separate concepts for each near-synonym. However, this would simply push the lexical choice task into a higher-level process (perhaps text planning) that chooses which semantic objects to include in the input to the system. But it has been shown that lexical decisions (including the choice of near-synonyms) and syntactic decisions are mutually constraining, and so must be made at the same level of processing (Danlos, 1987; Nirenburg, Lesser and Nyberg, 1989; Meteer, 1992; Horacek and Zock, 1993; Wanner and Hovy, 1996). This in turn, requires a more flexible linking between concepts and words.

4.2.2 Discrimination networks

The first approach to a flexible linking was Goldman's development of *discrimination networks* in his BABEL generator (Goldman, 1975). A discrimination net is a decision tree with words as leaves, and decision procedures (in Lisp code) as nodes: one simply traverses the tree running decision procedures at each node until a leaf is reached. In effect, a word is seen as a having a *core meaning* plus conditions on its use. For example, after choosing the primitive concept `ingest`, a discrimination net is traversed to determine if it should be realized as *eat*, *drink*, *breathe*, or *take* (medicine), depending on the substance ingested. Despite the fact that Goldman's decision procedures were arbitrary and informal and that the semantic concepts were perhaps too general—why would the input be so vague as to include `ingest` when clearly `eat` or `drink` is meant?—the general idea of using discrimination networks was influential in many future systems. It's not difficult to see that discrimination nets could be of use in choosing from among near-synonyms, if the semantic primitives were made more specific.

Hovy's PAULINE (Hovy, 1988), for instance, is able to choose near-synonyms according to syntactic, semantic, and pragmatic criteria (including stylistic goals such as formality, simplicity, and force) by using discrimination procedures indexed under a common concept. For example, under the concept `die` one finds *kick the bucket* (chosen if colloquial formality is desired) and *die* (chosen if neutral formality is desired). Or, to vary the affect of an expression (in order to express the speaker's opinions), PAULINE can choose "enhancing" or "mitigating" words. For instance, from the concept `hit` PAULINE accesses *hit* (as neutral) and then discriminates to *smash* or *slam* as enhancers and *tap* as a mitigator. Unfortunately, in lacking an overall theory of lexical meaning, the discriminations made are not principled.

A second major criticism of discrimination nets is that they encode lexical differences procedurally—the differences are in effect represented implicitly. A declarative representation is preferable for several reasons. First, the information about difference should be available to the other planning components of the generator for, say, reasoning about possible effects. This would allow a more flexible order in text generation. Second, a declarative representation is required so that the lexical resources can be used in different applications such as language understanding, machine translation, and text generation.

A final criticism of discrimination nets is that they are inefficient in representing multidimensional differences. If a set of words differs in several dimensions at once, then cross-classification would be required, or discrimination procedures would have to be duplicated in different parts of the network.

4.2.3 Lexis in systemic functional linguistics

Notwithstanding the above criticisms, discrimination nets have been extensively studied in Halliday's systemic functional linguistics (Halliday, 1994). In the theory, linguistic meaning is encoded in a *lexicogrammar*, which is a set of discrimination networks whose nodes are called systems. Generation proceeds by means of traversing the networks and making a decision at each system. During a traversal, a set of features is assembled, one for each system encountered, which is then converted into a syntactic structure by means of realization rules associated with each feature. But the traversal also makes lexical decisions because, in theory, "lexis is most delicate grammar" (Halliday, 1961); that is, there is no difference in the nature of lexis and grammar—they merely shade one into the other in terms of "delicacy", or precision.

The important point here is that a lexicogrammar is intended as a "principled account of those meanings which are formally expressible through language" (Tucker, 1995, p. 140)—which amounts to distinguishing between a level of *linguistic semantics* and *conceptual semantics*. Thus, when constructing a lexicogrammar, systemicists are careful to motivate the discriminations by identifying differences in linguistic form. For instance, one difference between *scatter* and *strew* is that the former can take a benefactor, as in *scatter the pigeons some bread crumbs*, and the latter cannot, motivating a system that discriminates on the basis of benefaction (Hasan, 1987).

In practice however, it is difficult to always find and motivate a formal difference in the use of one word over another, beyond the trivial difference that two words are different forms. For instance, *scatter* also implies a separation of parts, and a throwing that lets things lie where they may, whereas *strew* implies a spreading at intervals (Gove, 1973), which do not seem to have formal consequences. As a result, sentence generation systems based on systemic theory, such as Penman (Penman Natural Language Group, 1989), KPML (Bateman, 1996), and FUF/SURGE (Elhadad and Robin, 1996) make only very general discriminations about lexis (e.g., between mass and count nouns). In Penman, for instance, words are not really 'chosen', because semantic concepts in its input are usually linked directly to words. Because of this, such sentence generators are now usually employed solely as surface generation components, and are called upon after the words have been chosen (Stede, 1999; Elhadad, McKeown and Robin, 1997) during a sentence-planning process (see section 4.2.5, below).

Still, the prevailing view in systemic functional linguistics is that word choices ought to be made by the lexicogrammar (Hasan, 1987; Matthiessen, 1991; Martin, 1992; Tucker, 1995). Progress has been made, but it is evident that the principle behind the lexicogrammar is ultimately untenable when it comes to the representation of near-synonyms. On the one hand, if we obey the principle, then we must either ignore some (conceptual) differences between the words that might be important in text generation, or represent them outside of the lexicogrammar. On the other, making a discrimination in the lexicogrammar based on such differences would violate the principle and thereby produce a systemic network no different than the discrimination nets discussed above and thus subject to the same criticisms. As evidence, there is an apparent reluctance among systemicists to extend their networks to the point in delicacy where near-synonyms can be distinguished. Hasan, for instance, does not consider other near-synonyms of *scatter* such as *sow, broadcast, spread,* and *sprinkle.* Martin stops at a point leaving the verbs *name, call, christen, make, elect* and *dub* undifferentiated. And Tucker, in his extensive lexicogrammar of quality (usually realized by adjectives), does not provide systems for discriminating among the near-synonyms of *thin* (e.g., *slim, slender, gaunt,* etc.), of *brave,* of *famous,* or of any other relatively specific adjectives.

But not to completely diminish the contribution of systemicists to near-synonymy, I believe the important contribution they make is in two areas. One is in representing structural differences in word use including selectional preferences (e.g., *spill* takes a liquid object, *strew* a solid (Hasan, 1987)), subcategorization (Hasan, 1987; Martin, 1992), and collocational phenomena (Martin, 1992; Tucker, 1995) in a unified framework—these all involve the formal expression of meaning. The second is in providing a theoretical account of stylistic and attitudinal differences (e.g., *friend* vs. *mate*) in the same framework.

4.2.4 Simple frame representations

One alternative to discrimination nets is to represent lexical knowledge declaratively in frames. Nirenburg and his colleagues (Nirenburg and Nirenburg, 1988; Nirenburg, Lesser and Nyberg, 1989; Nirenburg and Defrise, 1992) developed DIOGENES, the first significant system for text generation and machine translation to use frames to represent lexical knowledge. For instance, the frame for *boy*, shown in figure 4.3, includes attributes for sex and age. Related words (e.g., *kid, teenager, youth, child, young man, schoolboy,* and so on) would be represented with the same attributes but with different values (although it is curious that some of these words are listed as synonyms in the entry). All of the attributes and values are defined in an ontology. Additionally, in a basic attempt to represent 'essentialness', each attribute is associated with an importance score that expresses the "saliency of the various relations for the identity of the entry head [i.e., the word in question]" (1988, p. 473). This marks a departure from a strict necessary-and-sufficient features approach. In later work, Nirenburg and Defrise (1992) add pragmatic (stylistic) factors such as *formality* and *colour*, modified from of Hovy's PAULINE.

With this representation, lexical choice is the problem of finding the word frame that

```
(make-frame boy
   (is-token-of (value person.CL))
   (sex (value male)
        (importance 10))
   (age (value (2 15))
        (importance 4))
   (lexeme (value "boy"))
   (para-collocation
      (synonym lad kid child)
      (antonym girl adult)
      (hypernym person)))
```

Figure 4.3: Lexical entry for *boy* (Nirenburg and Nirenburg, 1988).

matches an input frame (with desired values specified on each of the attributes). Of course, not all combinations of attribute–value pairs will be covered by word frames and the importance scores must be taken into account, so they use a proximate matching metric to choose the closest matching word frame. They claim that a lexical choice process requires such a "preference-assigning capability" to be able to make choices.

Two criticisms can be made of the representation itself. First, it is flat. That is, it cannot represent the complex (recursive) structures that are necessary for the encoding of some semantic nuances (see figure 4.1 above) or for paraphrasing (see the next section). Second, it can be difficult to motivate what attributes are necessary, their numeric values (especially of importance), and the associated matching functions; they are unprincipled and at worst arbitrary.

Furthermore, McDonald (1991) showed that applying a matching metric can be computationally expensive because, without any other means of indexing lexical items, every lexical item in the lexicon would have to be compared to the input. The same problems of inefficiency would arise if we were to use a taxonomic lattice or cross-classification (see sections 4.1.1 and 4.2.2). A strict taxonomy is an efficient index into the lexicon, whereas a lattice can be very inefficient. That is why Tucker (1995) stresses that cross-classification should be kept to a minimum in systemic networks for lexical choice (see section 4.2.3).

Nevertheless, the general ideas are important, because they recognize both the multidimensional nature of word meaning and that proximate matching is required for lexical choice. One area where the simple frame representations may be adequate is in representing the stylistic features of words. Stede (1993) uses a more rigorous and systematic approach, in contrast to the work on DIOGENES and PAULINE. Again based on Hovy's work, Stede identifies seven stylistic dimensions including formality, euphemism, and slant. Then psychological experiments would be used—he didn't perform them—to validate intuitions about the value of each stylistic dimension for a word. He defines a "stylistic preference function" that determines the most stylistically appropriate word given an input profile. For example, under a goal of producing **slang** style text (versus, say, **newspaper** style), *throw out* would be chosen rather than *evict*, and *pulverize* rather than *tear down*.

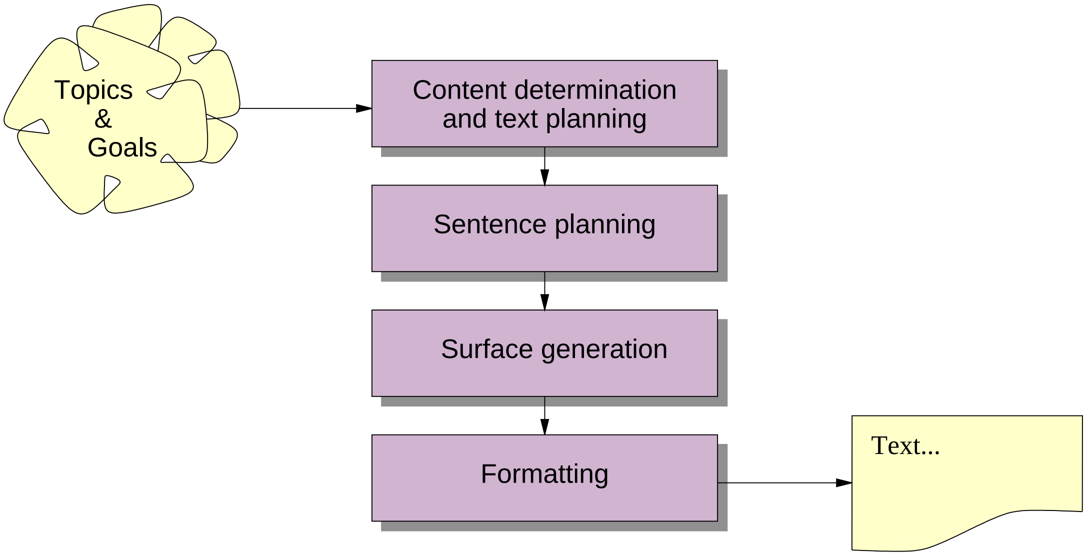


Figure 4.4: The 'consensus' architecture of current natural language generation systems (Reiter, 1994).

4.2.5 Sentence planning and paraphrasing

In recent years, a consensus about the overall architecture of NLG systems has emerged despite the sometimes conflicting theoretical claims made by the developers of the various systems (Reiter, 1994; Reiter and Dale, 1997). The 'consensus' architecture, depicted in figure 4.4, breaks up the process of generating text into a set of modules connected in a pipeline. The content-determination module maps the initial input of the system onto a conceptual form or text plan. This conceptual form is then passed to the sentence planning module,[5] which maps it onto a series of sentence plans. Finally, the surface generation and formatting modules realize the individual sentence plans as a formatted natural-language text.

The consensus architecture is very general. Researchers disagree on what the 'primitive' tasks of generation are, on where to break the modules (i.e., what tasks they should cover), on how the modules should communicate (e.g., in a uni-directional pipeline, or with feedback), on how the internal architectures of each module should be designed, and even on what the initial input to an NLG system should consist of. The RAGS project (Reference Architecture for Generation Systems), currently underway at ITRI, Brighton, will attempt to establish a more concrete standard for NLG systems.

In systems that employ an abstract language for representing sentence plans, rather than a less flexible 'template' approach, the purpose of the sentence generator is reasonably well-accepted, because the abstract languages tend to represent knowledge about sentences at a roughly similar level of abstraction. For example, the Penman (and KPML) sentence generator takes as input a sentence plan formulated in Sentence Plan Language (SPL) (Kasper, 1989). A plan is a typed-feature structure in which the types are defined in the Penman's Upper Model, an ontology of language-specific concepts that represents a language-specific linguistic classifi-

[5] The name 'sentence planning' was suggested by Rambow and Korelsky (1992), but it has also been called 'microplanning' (Panaget (1994), after Levelt (1989)).

cation rather than a deep conceptual classification. The Upper Model places constraints on the types of role and their fillers, that can be associated with a concept. For instance, a 'Directed-Action' can have an actee role, whereas a 'Nondirected-Action' cannot. The constraints ensure that sentence plans are well-formed (so that they can be realized as grammatical sentences). Input to the FUF / SURGE generator is on the whole formulated in a language similar to SPL, but the generator does make some different assumptions about the content of the input. (See Reiter and Dale (1997) for a more complete discussion of current sentence generation technology.)

Though it is not yet generally accepted, sentence planning has been converging on a kind of standard as a result of a current surge of research in the area. This concerns us, because lexical choice is now considered one of the main components of sentence planning (the others include choosing grammatical relations, generating referring expressions, ordering the constituents, and grouping propositions into clauses and sentences). In fact, in many systems the lexicon has taken on the central role as a bridge between a conceptual (language-neutral) input and a linguistic output (e.g., ADVISOR II (Elhadad, McKeown and Robin, 1997), MOOSE (Stede, 1999), and Hunter-Gatherer (Beale et al., 1998)). That is, individual lexical entries associate configurations of concepts (i.e., denotations) with partial semantic / syntactic structures. Figure 4.5 gives an example from the MOOSE system. This approach can be called *lexicon-driven* sentence-planning, because all processing centers on choosing words. Since the denotation of a word encodes the necessary applicability condition of the word, a set of words is chosen whose denotations collectively cover the input without redundancy, and whose partial syntactic structures can be assembled into a well-formed syntactic structure. While the computational methods may be different—ADVISOR II uses functional unification, MOOSE uses pattern-matching and searching, and Hunter-Gatherer uses constraint satisfaction—the results are the same: the systems can all produce a wide variety of *paraphrases* of the same input. For example, figure 4.6 gives an example of paraphrasing from the MOOSE system.

The above systems, either implicitly or explicitly, are based on a theory of lexical semantics that posits two levels of representation: a *syntactic–semantic* level and a *conceptual–semantic* level. The former level is, obviously, language-specific, and accounts for compositional meaning, that is, it links the semantic participants of a situation to surface level syntactic positions so that words can be combined into syntactically well-formed sentences.

The latter level is responsible for the building up of language-independent representations of text that can be reasoned with. Nirenburg and Levin (1992) argue that a clear separation between ontological and syntactic levels is required for multilingual applications. But it is also necessary in order to account for various phenomena in a single language including regular polysemy (Pustejovsky, 1995) and verb alternation patterns (Stede, 1998).

One alternative to the lexicon-driven approach is a task-oriented, distributed approach. In HealthDoc (DiMarco et al., 1995), the sentence planner is divided into a number of modules that perform particular sentence planning tasks (including discourse structuring, content delimitation, aggregation, reference planning, and lexical choice). The modules run in parallel and communicate by means of a central blackboard controlled by an administrator module (Wanner and Hovy, 1996). The architecture is theoretically appealing, because sentence planning tasks each require different knowledge sources and algorithms, suggesting modularization, yet are highly interdependent, requiring parallel execution. Modularization also gives each task (i.e., module) the opportunity to make explicit sentence planning decisions, which is not possible in the lexicon-driven approach.

Unfortunately, the blackboard architecture for sentence planning has not yet been developed to an extent sufficient to demonstrate whether or not it is a practical approach to sentence planning. It might turn out that it allows too much flexibility, requiring very complex inter-

Lexical entry for *remove*:

```
Denotation: (event (PRE-STATE (V1 location-state (LOCATUM A)
                                                 (LOCATION B)))
                   (ACTIVITY (V2 (CAUSER C)))
                   (POST-STATE V3 location-state (LOCATUM A)
                                                 (LOCATION (not B))))
Covers:     (event V1 V2 V3 (not B))
PSemSpec:  (X / directed-action :lex "remove"
              :actor C :actee A :source B)
```

Lexical entry for *uncork*:

```
Denotation: (event (PRE-STATE (V1 location-state (LOCATUM (cork A))
                                                 (LOCATION (bottle B))))
                   (ACTIVITY (V2 (CAUSER C)))
                   (POST-STATE V3 location-state (LOCATUM (cork A))
                                                 (LOCATION (not B))))
Covers:     (event V1 V2 V3 A (not B))
PSemSpec:  (X / directed-action :lex "uncork"
              :actor C :actee B)
```

Figure 4.5: The lexical entries for *remove* and *uncork* in the MOOSE system. Most sentence planners represent similar information in their lexical entries. The denotations are configurations of concepts (lowercase) and relations (uppercase) defined in an ontology, and variables. The 'covers' component is a list of nodes in the denotation that the word actually covers. Notice that *uncork*, a specialization of *remove*, covers the variable `A`, for the cork, whereas *remove* does not. The 'PSemSpec' component is a partial syntactic template in Sentence Plan Language (SPL).

Input to the sentence planner (a SitSpec):

```
(event-1 (PRE-STATE  (open-state-1 (VALUE 'closed)
                                   (OBJECT bottle-1)))
         (ACTIVITY   (move-1 (CAUSER jill-1)
                             (OBJECT cork-1)
                             (PATH (path-1 (SOURCE bottle-1)))))
         (POST-STATE (open-state-2 (VALUE 'open)
                                   (OBJECT bottle-1))))
```

Paraphrases output:

```
Jill opened the bottle.
Jill uncorked the bottle.
Jill removed the cork from the bottle.
The bottle was opened by Jill.
The bottle was uncorked by Jill.
The cork was removed from the bottle by Jill.
```

Figure 4.6: An example of paraphrasing in the MOOSE system. The 'SitSpec' represents a specification of the situation to be verbalized (uppercase terms are relations, lowercase terms are instances of concepts, and quoted terms are values).

module conflict resolution strategies. Moreover, lexical choice, as a sentence planning task, cuts across many of the other tasks, including reference planning, discourse structuring (e.g., choosing discourse markers), and aggregation, so it is difficult to isolate in one module.

Now, regardless of which approach one takes to sentence planning, it must involve paraphrasing, and one way to paraphrase a sentence is to replace one or more of its words with near-synonyms. However, in the above systems, near-synonyms are generally treated as absolute synonyms; they are given the same denotations modulo selectional restrictions. Thus, the systems can enumerate the different paraphrases, but not actually choose *which* paraphrase is the most appropriate. There are, however, a few exceptions to this sweeping statement.

In Stede's MOOSE (Stede, 1996), it is possible to represent stylistic differences (what he calls connotations) between near-synonyms. Stede incorporated his "stylistic preference function", mentioned above, into MOOSE. Denotationally, though, it is not possible to represent most of the fine-grained denotational distinctions that can exist between near-synonyms, because denotations are treated as necessary applicability conditions.

Elhadad's ADVISOR-II (Elhadad, McKeown and Robin, 1997) can realize an "argumentative evaluation" of a participant of the situation by choosing appropriate lexical items. For example, when the system would normally choose the verb *involve* to express the relation between a university class (i.e., a course) and its assignments (e.g., "the AI class *involves* seven programming assignments"), it can instead choose *require* to evaluate the course as difficult: (e.g., "the AI class *requires* seven programming assignments"), because *require* can convey an "evaluative connotation" of difficulty that *involve* cannot.

Figure 4.7 shows the lexical entries of *involve* and *require* in the ADVISOR-II system. Both words denote an attributive process that relates a class to its assignments, as indicated by the 'participants' feature. However, the entry for *require* contains additional information that indicates that it evaluates the class as high on the scale of difficulty. (Note that small numbers in

$$\left[\begin{array}{ll} \text{process} & \left[\begin{array}{l} \text{type attributive} \\ \text{lex } \textbf{\textit{“involve”}} \end{array}\right] \\ \text{participants} & \left[\begin{array}{ll} \text{carrier} & \left[\text{class } \boxed{1}\right] \\ \text{attribute} & \left[\text{assignments } \boxed{2}\right] \end{array}\right] \end{array}\right] \qquad \left[\begin{array}{ll} \text{process} & \left[\begin{array}{ll} \multicolumn{2}{l}{\text{type attributive}} \\ \multicolumn{2}{l}{\text{lex } \textbf{\textit{“require”}}} \\ \text{ao} & \left[\begin{array}{l} \text{evaluated } \boxed{1} \\ \text{scale } \left[\text{name “difficulty”}\right] \\ \text{conveyed verb\_level} \\ \text{orientation } + \end{array}\right] \end{array}\right] \\ \text{participants} & \left[\begin{array}{ll} \text{carrier} & \left[\text{class } \boxed{1}\right] \\ \text{attribute} & \left[\text{assignments } \boxed{2}\right] \end{array}\right] \end{array}\right]$$

Figure 4.7: The lexical entries, modified for clarity, of *involve* and *require* in ADVISOR-II (Elhadad, McKeown and Robin, 1997).

boxes indicate co-referential terms, so in the entry for *require*, it is the class that is being evaluated.) Crucially, the evaluation is not a necessary applicability condition; the ‘ao’ feature, for “argumentative orientation”, indicates this. *Require* can be chosen whether or not the input asks for difficulty to be expressed; however, when difficulty is desired *require* would be chosen instead of its near-synonyms (unless difficulty can be expressed in some other way, e.g., by explicitly saying so, or by choosing an appropriate modifier: “AI has *many* assignments”). Thus, ADVISOR-II can represent and use some types of fine-grained meaning akin to implications (which we discussed in section 3.5.3). But the model is very preliminary and ad hoc; for instance, the lexical entries are highly tailored to the student-advisory domain (*require* does not always imply difficulty, in general usage). And, it is not evident how terms such as “difficulty” (in figure 4.7) are defined, or how more complex multi-dimensional differences can be represented. However, Elhadad, McKeown and Robin (1997) did formalize the idea of a *floating constraint*, which is a constraint on sentence planning that can be satisfied by different types of decision and simultaneously with other constraints. In the above example, the desire to express “difficulty” is specified as a floating constraint, so it can satisfied in different ways (e.g., with an adjective, an intensifier, a connotative verb, and so on). The need for the ability to explicitly specify floating constraints (i.e., differently from other constraints) in ADVISOR-II is an artifact of the architecture of its sentence generator, FUF/SURGE. Nevertheless, the idea of a floating constraint is very useful, as we will see in section 9.2 when we discuss the input to our lexical choice system. Briefly, the input must be able to specify *preferences*, similar in spirit to floating constraints, that can be satisfied collectively and simultaneously by several different types of lexical and syntactic choice.

Finally, in Hunter-Gatherer (Beale et al., 1998), word meanings are represented as in MOOSE and ADVISOR-II, but the lexical choice process can ‘relax’ its semantic matching constraints by assigning varying penalties to mismatches. Near-synonyms are not explicitly accounted for, but they could be represented with slightly different denotations, which would each match the input specification to different degrees. While this moves away from denotation as strict necessary-and-sufficient features, it does so at the expense of a declarative representation of the differences. That is, the denotations themselves still represent necessary-and-sufficient features; it is the matching procedure that finds the differences.

4.2.6 Ward's connectionist model

Ward (1994) developed a radically different approach to text generation. In his connectionist model, implemented as FIG, he uses a single network to represent all conceptual, semantic, syntactic, and lexical knowledge. Thus, nodes can represent concepts, relations, words, and syntactic categories. The input to FIG activates a number of nodes in the network. Then, after each successive application of spreading activation over the network, the most activated word is output, thereby building a sentence left to right. For example, if the concept nodes for `woman`, `go`, `Thursday`, and `river` are currently activated, and the "syntactic considerations" are currently activating verbs, then activation would spread to the word *go* and it would be output. So the model considers all of the options in parallel and takes the best choice.

This model has some interesting ramifications for the representations of words. First, there is no notion of necessary and sufficient conditions for defining a word. Every word is linked to a concept node that can be in turn linked to a set of other concept nodes that represent 'features' of the concept. For instance, the word *boy* is linked to the concept `boy`, which is in turn linked to `male` and `child`. So, if either `male` or `child` is activated, some activation will pass eventually to *boy*. However, the model is somewhat simplistic because not only are features conflated with concepts but there is no distinction between essential, typical, and peripheral features as all features contribute equally to the meaning of a word, though weights assigned to links may encode some of this knowledge.

Second, near-synonyms are handled by having a concept link to several words, e.g., `river` links to both *river* and *stream*, one of which is assigned default status by an a priori weighting of the links. Theoretically, the model could represent semantic distinctions between the near-synonyms using the method above. For example, the concept `destination` links to both the preposition *to* (the default) and *into*. If the concept `region`, is activated at the same time as `destination`, then *into* is chosen rather than the default *to*, because `region` also links to *into*; however, this presumes a careful tuning of the weighting factors. In practice, though, Ward does not explore this theory and instead differentiates near-synonyms according to only their collocations with other words. Actually, the connectionist model handles collocations easily, because they are represented simply as links between the associated words; when one is activated, the other receives activation as well.

In any case, the connectionist model would seem to be a valid approach to representing some sets of near-synonyms, but this has not yet been explored.

4.2.7 Statistical approaches

However well a knowledge-based approach to MT or NLG works in particular cases (Nirenburg et al., 1992), the systems are not robust in general because of gaps in knowledge, which are inevitable because of the enormous volume of knowledge that needs to be represented (Knight et al., 1995). Statistical approaches, on the other hand, can be very robust, because they use statistical models of individual languages (for both analysis and generation) and of bilingual lexical correlations between languages (for translation) that can encode, shallowly, a broad range of linguistic phenomena.

Brown et al. (1990; 1993), in their seminal work on statistical MT, take the view that every sentence in one language is a possible translation of a sentence in another; thus, translating from, say, French to English is the problem of searching for the English sentence E that maximizes $\Pr(E|F)$ (the probability that a translator will produce E when given F), for a given French sentence F. Since they cannot accurately estimate this probability on its own, they apply

Bayes's theorem and arrive at the "Fundamental Equation of Machine Translation:"

$$\hat{E} = \operatorname*{argmax}_{E} \Pr(E) \Pr(F|E) \tag{4.1}$$

Thus, finding the best translation $\hat{E}$ of F—and the best words—involves two factors: the probability $\Pr(F|E)$ of it being translatable into F and the probability $\Pr(E)$ of it being 'well-formed' (cf. Knight and Hatzivassiloglou's (1995) "fluency") regardless of its connection to F. The former is called the translation model, and the latter the language model. Thus the overall translation task is partitioned into a transfer task (finding the words) and a generation task (putting them together).

Near-synonyms enter the model implicitly in estimating both $\Pr(F|E)$ and $\Pr(E)$. Brown et al. (1993) investigate several models for estimating the former, all of which involve the probabilities of individual word-to-word translations. Essentially, a bilingual lexicon must be constructed in which each pair of equivalent terms is assigned a probability of translation.[6] In figure 3.6 (page 46), for instance, we would have to assign to each link leaving word a probability that the word translates to the word on the end of the link. Brown et al. automatically extracted word-to-word translation probabilities from the *Hansards* (the bilingual proceedings the Canadian parliament), and found, for instance, that the English *answer* has a probability of 0.442 of translating to *réponse*, 0.233 to *répondre*, 0.027 to *solution*, and so on. Though they don't give the data, it is likely that *response* also has a high probability of translating to *réponse*, and likewise for *reply* and other near-synonyms. Thus, turning it around, *réponse* could be translated to *answer*, *response*, or *reply* with different probabilities. Yet, by using only a bilingual lexicon, the same, most probable, word would always be chosen. So it falls to the language model to make the determination of which word is best in context.[7]

A language model, in theory, can tell you the probability of a single word occurring given all of the words that precede it in a sentence. So, given a set of near-synonyms as options to fill a given position in a sentence, one would choose the most probable near-synonym according to all of the preceding words. In practice, though, it is too computationally expensive to use such a complex language model (that looks back so far), so usually at most two previous words—the trigram model—are used. While Brown et al. show that a trigram model is enough to give a properly structured sentence in many cases, one suspects that it is not enough to differentiate between near-synonyms (Edmonds, 1997). Knight and Hatzivassiloglou (1995) and Langkilde and Knight (1998) show that such a model can also be used in text generation. Their bigram implementation is able to account for some aspects of style and for short-range collocations.

In the end, it is not clear whether a statistical approach is capable of accounting for fine-grained differences in meaning—it certainly can't account for explicit (knowledge-based) differences. Its strength, rather, lies in determining which near-synonym is most typical or natural in a given context. In fact, near-synonyms seem to be of little concern in this research: Brown et al. view such variations as a matter of taste; in evaluating their system, two different translations of the same source that convey roughly the same meaning (perhaps with different words) are considered satisfactory translations.

[6] Brown et al. spawned an area of research in automatically aligning parallel bilingual texts by sentence and by word for the purpose of automatically building bilingual lexicons (Gale and Church, 1993; Kay and Röscheisen, 1993; Kumano and Hirakawa, 1994; Smadja, McKeown and Hatzivassiloglou, 1996; Melamed, 1997).

[7] Of course, bilingual or multilingual dictionaries have always been used in translation, but like all dictionaries, they leave a lot of information unsaid. A human translator is usually able to use the context and her intuition to decide which of several alternatives is the most appropriate in any particular situation.

4.3 Conclusion

It should now be clear that near-synonymy is not adequately accounted for in the many theories of lexical semantics, nor in their implementations in computational systems. The conventional model of the lexicon (a synthesis of the decompositional and relational models) used in most NLG and interlingual MT systems fails in the following respects:

1. It entails an awkward taxonomic proliferation of language-specific concepts at the fringes of the ontology, thereby defeating the purpose of a language-independent ontology.

2. A taxonomic hierarchy also emphasizes hyponymy, backgrounding all other relations, which appear to be more important in representing the multi-dimensional nature of fine-grained word meaning.

3. It defines words in terms of only necessary and sufficient conditions, so it cannot account for indirect expressions of meaning and context-dependent meanings.

4. It is primarily feature-based, thus it cannot account for fuzzy differences or the complex interrelationships of the aspects of a word's meaning.

5. It is purely denotational, thus it cannot be used to represent expressed attitudes or style.

The contextual and prototype theories, while addressing some of these problems theoretically (e.g., indirect meaning and context-dependent meaning), have not matured to the point where they can be implemented in a knowledge-based computational system—possibly because they are founded on an explanation of how words interact with context and personal experience, which is not yet well understood. However, some of the aspects of these theories can be implemented using probabilities or other quantitative mechanisms.

Therefore, since we want to develop a model for representing near-synonyms that can be used in natural language applications, we need to develop a theory of fine-grained lexical meaning that accounts for as many of the phenomena of lexical variation as possible, yet is implementable in a computational system. We must avoid, or move beyond, the kind of representation suggested by Big Brother's dictum, given at the start of this chapter, which is so enticing computationally, yet so restrictive representationally. The logical direction to take is to start with a traditional theory (which gives rise to the ontological model), and to modify or extend it with aspects, at least in spirit, of the contextual and prototype theories of meaning. We address this idea in the next chapter.

Chapter 5

A theory of fine-grained lexical meaning

This chapter proposes a theory of fine-grained lexical meaning that is intended to account for many of the phenomena related to near-synonymy that we observed in chapter 3. The theory builds on and improves various of the current theories of lexical semantics, but was inspired by the architectural and descriptive principles that lexicographers use in developing thesauri and synonym dictionaries. The central motivation is that near-synonyms cluster together naturally, because they are so close in meaning relative to other words. And it's not just similarity in meaning, but similarity in *essential* or *core* meaning that pulls near-synonyms together (see our definition of near-synonymy in chapter 2). Lexicographers have given various names to their sets of synonyms, including 'clusters' (*Roget's International Thesaurus*), 'lists' (*Webster's Collegiate Thesaurus*), 'groups' (*Webster's New Dictionary of Synonyms*), and 'synsets' (WordNet), but we prefer the term 'cluster', because it implies a distinguishable group within a larger whole (see Gove's (1973) usage note for *cluster*). In the next section we discuss the shortcomings of the current theories, and then propose how we can overcome these shortcomings with a new theory of fine-grained lexical meaning.

5.1 Motivating a new theory

As we discussed in chapter 4, the very existence of near-synonymy (and of polysemy) makes evident the inadequacies of the theories of lexical meaning that are traditionally used in natural language systems (i.e., those theories which give rise to decompositional or ontological models). Clearly, the meaning of a word is not static and fixed *a priori* (disregarding changes over time), but is fuzzy and context-dependent. So, clearly, words cannot be defined solely in terms of truth-conditions, nor should they be linked in a one-to-one relationship with representations of concepts. Nevertheless, the traditional theories have remained popular, because they have been relatively easy to implement, and have been successful in many systems to date, which had no need for fine-grained semantics.

In particular, the now-conventional knowledge-based model of the lexicon represents the meaning of each word sense by linking it to a separate concept, or configuration of concepts, defined in an ontology. A knowledge-based model of this sort appears to be the most promising and logical starting point to developing a new, more comprehensive theory of near-synonymy.[1]

[1] A knowledge-based model is of course not the only possible starting point. For instance, in chapter 6 we de-

Hence, in order to motivate a new theory of fine-grained lexical meaning, we will examine how this model copes with the problems of representing the fine-grained meanings (and differences between the meanings) of near-synonyms. This will serve to highlight problems with the model with respect to near-synonymy in the following three areas:

- The adequacy of coverage of phenomena related to near-synonymy.
- Engineering advantages, both in the design of an efficient and robust lexical choice process, and in the design of lexical entries for near-synonyms.
- The tractability of reasoning about concepts during natural language understanding and generation.

Let's begin with an attempt to represent the meanings of the English near-synonyms of *error* in a hierarchy. First of all, it's not clear that one can even come up with a reasonable hierarchy for these words. Is a *mistake* a kind of *error*, or vice versa? Or, perhaps each denotes a subconcept of a third, more generic, concept. Figure 5.1 shows one plausible hierarchy that takes this view by introducing the concept of `Generic-Error`. The main distinction made is between blameworthy errors and non-blameworthy errors. As explicated in Gove's usage note (see figure 1.1), the distinction should be continuous rather than discrete, but it is not possible to represent variation on a continuous dimension in a taxonomy. Furthermore, the *error* words also differ in their implication of varying levels of severity of criticism, but this is not represented in this taxonomy, nor can it be without introducing cross-classification. There are still more problems with this taxonomy. For instance, a *faux pas* should really be represented as both a `Social-Error` and a `Stupid-Error`, but this would also require cross-classification. And while *blooper* is represented as a kind of *blunder*, this is not entirely correct, since *bloopers*, while stupid, are not necessarily as serious as blunders. (So a separate distinction between serious versus amusing stupid-errors might be necessary.)

The problem is that the concepts denoted by the near-synonyms are not generally related to each other by specificity, the central organizing principle of the taxonomy. At best, this hierarchy can be thought of as one individual's conceptualization of the meanings of the *error* near-synonyms. At worst, the hierarchy is simply artificial.

We can overcome some of the coverage problems of this representation by using cross-classification, or a taxonomic lattice, which amounts to using a kind of feature-based representation of concepts. That is, a word such as *blooper* could be classified under the concepts (or features) of `Generic-Error`, `Stupid-Error`, `NonBlameworthy-Error`, and `Amusing-Error`; *blunder* under `Generic-Error`, `Stupid-Error`, `Blameworthy-Error`, and `Serious-Error`; and so on for the other near-synonyms. Such a scheme appears to be a good way of representing fine-grained denotational distinctions, but it gets more complicated as we move up into the more general levels of the ontology, which in turn could lead to a more complex and possibly inefficient lexical choice process. As we mentioned in section 4.2.4, it is beneficial to keep cross-classification to a minimum. This would be possible if we could concentrate it at the extremities of the ontology, as in this example, but unfortunately we cannot, since sets of near-synonyms at all levels of the ontology can have multi-dimensional differences. And even the most general of concepts can be lexicalized with various near-synonyms.

Consider what happens when we apply cross-classification at the more general levels of an ontology. For instance, the concept `Object`, usually at or near the root of an ontology, can be

velop a statistical model.

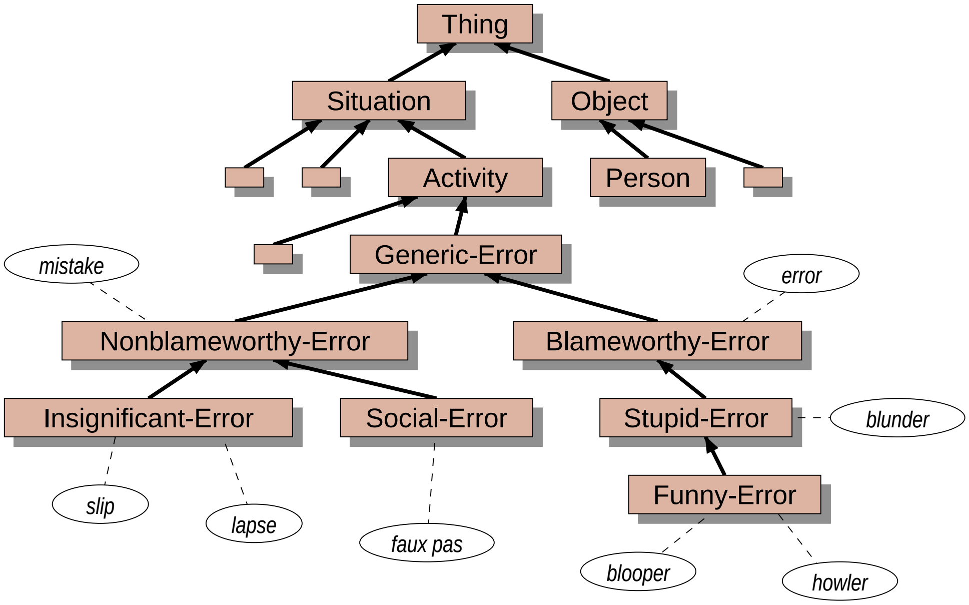


Figure 5.1: A possible taxonomy for the near-synonyms of *error*.

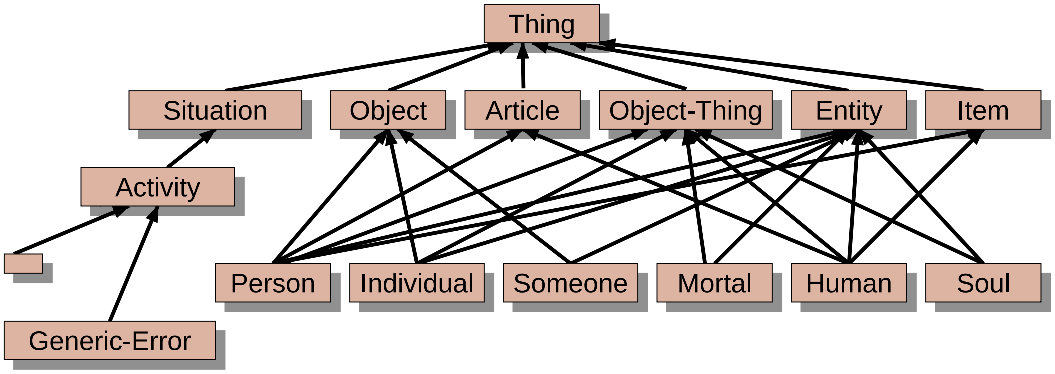


Figure 5.2: A lattice of concepts near the top of the ontology.

lexicalized as *object, thing, entity, item, article*, and so on.[2] If each of these five words were given its own concept and they were taxonomized then how should we attach their subordinate concepts? Does a subordinate concept, say `Person`, attach just to one of the concepts, to some of them, or to all of them? And then the concept of `Person` itself has many slightly different lexicalizations (including *person, individual, someone, somebody, mortal, human, human being, soul*, and so on), which would each require separate concepts in this model. Figure 5.2 shows how quickly the complexity can multiply, and it involves only two coarse-grained concepts, `Object` and `Person`. Not only would lexicographers, and others, find it difficult to manage such a taxonomy (when they have to add new concepts and lexical entries to it), but the taxonomy could lead to intractable reasoning processes, as it would require first-order logic as an underlying formalism (see section 7.2 for more on intractability).

Finally, as we already discussed in section 4.1.1, even if we were to disregard these engineering and tractability issues, such a model could still not address the other phenomena related to near-synonymy. The model cannot account for the manner in which denotational meanings can be expressed; nor can it account for fuzzy distinctions, and distinctions involving style and attitude.

The preceding discussion shows that most of the problems occur when we attempt to make very fine-grained distinctions between concepts in the ontology. A better model of lexical meaning would somehow separate out the fine-grained distinctions from the ontology, or compartmentalize the representation of the distinctions, making them more manageable. We still want to retain the basic ontological model at some level of granularity (since it has proven useful in representing denotational aspects of word meaning), however we want to avoid positing a separate concept for every word. There are at least two possible solutions. The first is to treat near-synonyms as absolute synonyms, avoiding fine-grained representations altogether. Of course, this is not a solution because it simply denies the existence of near-synonymy.[3] The second is to cut off the ontology at a coarse grain, and to devise a distinct subconceptual level for the fine grain.

5.2 Subconceptual clusters of near-synonyms

Our theory is that the meaning of a word, however it manifests itself in text or speech, *arises* out of a context-dependent combination of a basic inherent context-independent denotation and a set of explicit differences to its near-synonyms. (We don't rule out other elements in the combination, but these are the main two.) Thus, word meaning is not explicitly represented in the lexicon, but is created (or generated, as in a generative model of the lexicon (Pustejovsky, 1995)) when a word is used (i.e., during processing by a system). This theory preserves some aspects of the traditional theories—the basic denotation can be modelled using an ontology—but the rest of a word's meaning relies on other nearby words and the context of usage. And since the relations of a word to its near-synonyms are so important to its meaning—possibly more important than other relations—near-synonyms form *clusters*.[4]

The theory is built on the following three premises, which follow from our definition of

[2] These words vary in their range of application: *thing* can apply to something that is directly known or to something inferred from its effects, *object* specifically to something that is external to the mind, and *article* to something that is thought of as a member of a group (Gove, 1973).

[3] Treating near-synonyms as absolute synonyms might in fact be viable for many natural language processing applications, but not when fine-grained meanings are important, as in high-quality machine translation.

[4] It is very probable that synonym clusters could be built automatically by applying statistical techniques, such as cluster analysis, on large text corpora. For instance, Church et al. (1994) give some results in this area.

near-synonymy (in chapter 2) and our observations of the nature of near-synonymy (in chapter 3). First, the meaning of any word, at some level of granularity, must indeed have some inherent context-independent denotational aspect to it—if it did not, one would not be able to define or understand a word in isolation, as one in fact can. Second, nuances of meaning, while difficult or impossible to represent in positive, absolute, and context-independent terms, can be represented as differences, in Saussure's sense, between near-synonyms. That is, every nuance of meaning that a word might have can be thought of as a relation between the word and one or more of its near-synonyms. And third, differences must be described not as simple features or truth-conditions, but by structures that encode degrees of necessity, fuzziness, and relations to the context.

For example, the word *forest* denotes a geographical tract of trees at a coarse grain, but it is only in relation to *woods*, *copse*, and other near-synonyms that one can fully understand the significance of *forest* (i.e., that it is larger, more wild, etc.). Similarly, the word *mistake* denotes any sort of action that deviates from what is correct and also involves some notion of criticism, but only in relation to *error* and *blunder* does one see that the word can be used to criticize less severely than some alternative *error* words allow. None of these differences could be represented in absolute terms, because we would have to define some absolute notion of size, wildness, or severity, which seems implausible. So, at a fine grain, and only at a fine grain, we make explicit use of Saussure's notion of contrast in demarcating the meanings of near-synonyms. Hence, the theory holds that near-synonyms are explicitly related to each other not at a conceptual level but at a *subconceptual* level—outside of an ontology. In this way, a cluster of near-synonyms is not a mere list of synonyms; it has an internal structure that encodes fine-grained meaning as differences between lexical entries, and externally, it is situated within a conceptual model (i.e., the ontology) on one side, and a linguistic model on the other side.

Though the theory is still preliminary, it also takes into account the context of usage. Since the actual nuance conveyed when a word is used depends on the context, it is crucial in the theory that meaning is thought of as *arising* (or being generated) out of basic parts, rather than being actually represented in the lexicon.

Finally, the theory can also include a prototype account of word meaning, because there is no requirement in the theory that clusters of near-synonyms be static or fixed *a priori*. So each cluster, in effect, can act as a category that has graded membership and fuzzy boundaries, and therefore should exhibit prototype effects (Lakoff, 1987). But, as we said in section 4.1.4, more research is required so that we can come to a more complete understanding of the prototype effects involved in categories of near-synonyms. So, in formalizing the theory (see below), we have assumed that clusters of near-synonyms represent static and non-fuzzy categories; however, internal to clusters, we were able to bring in aspects of typicality of usage (see chapter 6), and components of meaning that are fuzzy and non-necessary (see chapter 7).

5.3 Three levels of representation

Specifically, the theory posits an additional level of semantic representation. Current computational theories suggest that two levels of representation, a *conceptual–semantic* level and a *syntactic–semantic* level, are necessary to account for various lexico-semantic phenomena in computational systems (see section 4.2.5). We believe a two-level semantics is also necessary, but to account for fine-grained meanings and near-synonymy, our theory postulates a third intermediate level (or a splitting of the conceptual–semantic level). The three levels are the following:

- A conceptual–semantic level.
- A subconceptual / stylistic–semantic level.
- A syntactic–semantic level.

At the top level, as in the current theories, we represent the coarse-grained basic, or essential, denotational meaning of a word. At the intermediate level, we represent the fine-grained context-dependent differences in word meaning. At the bottom level, we represent how a word can be combined with other words both syntactically and collocationally.

So, if we were to take the conventional ontological model as a starting point (in further formalizing the theory), we would cut off the ontology at a coarse grain and cluster near-synonyms under a shared concept rather than linking each word to a separate concept. Thus, on the conceptual–semantic level, a cluster has a *core denotation* that represents the essential shared denotational meaning of its near-synonyms. On the subconceptual / stylistic–semantic level, we represent semantic, stylistic, and expressive *distinctions* between the near-synonyms within a cluster in terms of *peripheral concepts* (defined in terms of concepts in the ontology), and stylistic and expressive (i.e., attitudinal) dimensions. On the syntactic–level, syntactic frames represent how words can be combined with others to form sentences. We will briefly discuss each of these terms after the following example.

Figure 5.3 shows one model of how the near-synonyms of *error*, *order*, *person*, and *object*, in several languages, would be represented according to this theory. In the figure, each set of near-synonyms forms a cluster linked to a coarse-grained concept, `Generic-Error`, `Generic-Order`, `Person`, and `Object`, respectively. The rectangles in the top part of the figure are concepts defined in the ontology (and are related by inheritance). Clusters in different languages are shown as groups of interrelated lexical entries that connect to the ontology. Thus, the core denotation of each cluster is the concept to which it points. Within each cluster the near-synonyms are differentiated at the subconceptual / stylistic level of semantics, as indicated by dashed lines that link the words in the cluster. (Note that the actual differences in terms of peripheral concepts, stylistic dimensions, and so on, are not shown in this figure.) Not all words in a cluster need be differentiated, and each cluster in each language can have its own 'vocabulary' of peripheral concepts, stylistic dimensions, and attitudinal dimensions for differentiating its near-synonyms. Note that the figure does not show a representation on the syntactic–semantic level. Below, we elaborate on each of the concepts just introduced.

Core denotation

As we said above, the core denotation of a cluster of near-synonyms is the basic inherent context-independent (and in this formulation of the theory, language-neutral) denotation shared by all of the near-synonyms. Figure 5.3 shows the core denotation in simplified terms, for in reality it can consist of an arbitrary structure of concepts. The core denotation is analogous to the concept to which a word links, in the conventional ontological model. Thus, at a coarse grain of semantic representation, this model is equivalent to the conventional model. That is, if we were to disregard the fine-grained differences between near-synonyms, then it would be as if the near-synonyms were absolute synonyms, all linked to the same concept. The main difference between this model and the conventional model is the subconceptual level of representation.

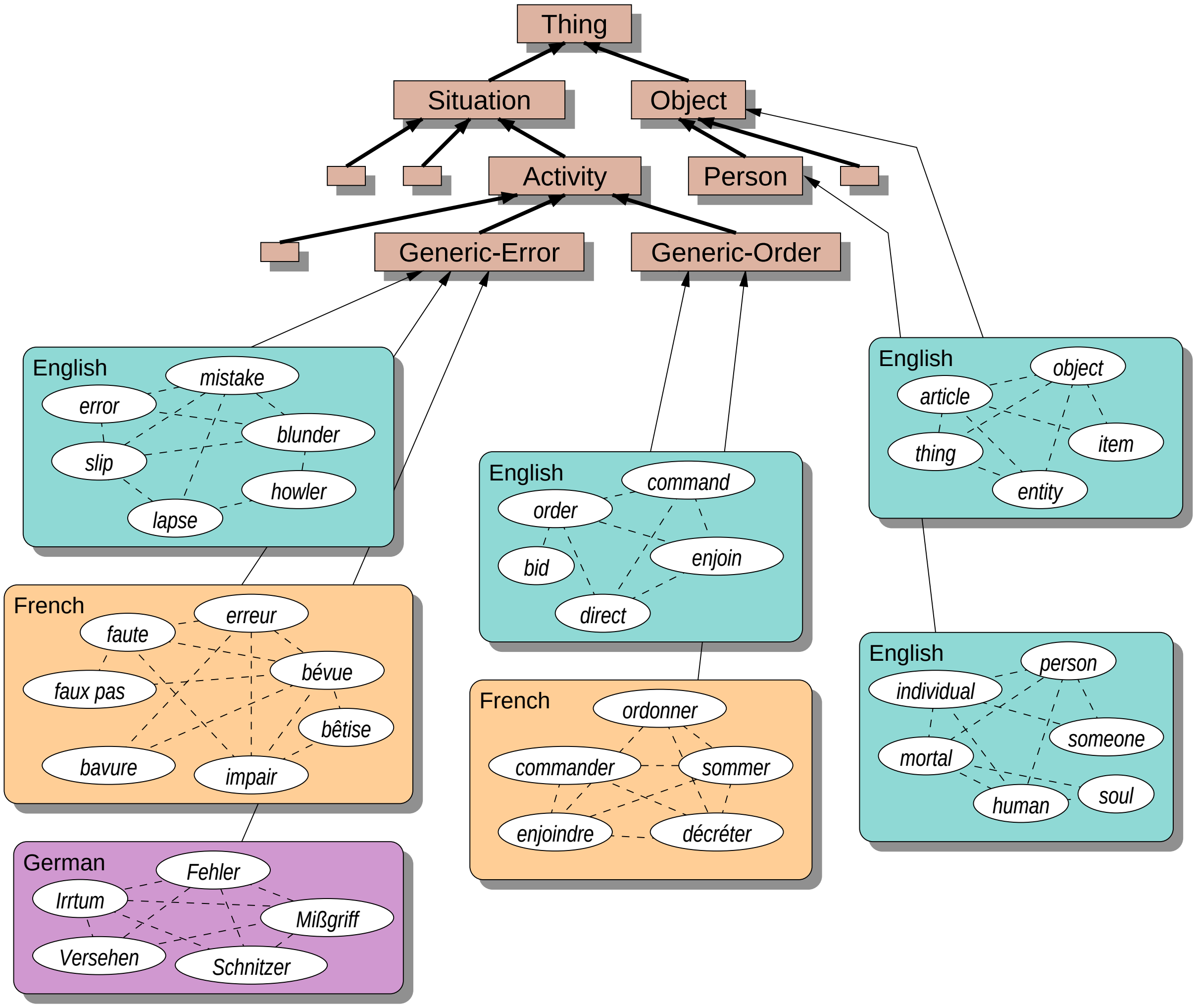


Figure 5.3: A clustered model of lexical knowledge.

Peripheral concepts

Informally, peripheral concepts form the basic vocabulary of fine-grained denotational distinctions. For instance, in differentiating the *error* words, a lexicographer would first decide that the basic peripheral concepts required might be ‘stupidity’, ‘blameworthiness’, ‘criticism’, ‘misconception’, ‘accidentalness’ and ‘inattention,’ among others (see section 3.7). Then the lexicographer would proceed to distinguish the near-synonyms in terms of these concepts, for instance, by specifying that *mistake* involves a less severe form of criticism than *error*.

More formally, peripheral concepts are structures of concepts defined in the same ontology as core denotations are defined in. In fact, every peripheral concept in a cluster must be in some sort of relation to the core denotation, because, after all, peripheral concepts represent ideas related to the core meaning of a cluster of near-synonyms. But since peripheral concepts have a different status than the core denotation, they can be used to represent non-necessary and indirect aspects of word meaning. They are concepts that might be implied, suggested, emphasized, or otherwise, when a word is used, but not always.

Distinctions between near-synonyms

Though near-synonyms can be distinguished in many ways, the subconceptual/stylistic–semantic level is particularly suited to account for variation in three out of the four broad categories into which we classified lexical variation in section 3: denotational, stylistic, and expressive variation. (Variation in the fourth category, collocational variation, can be handled on the syntactic–semantic level.) Details of how we can represent each type of distinction are given in chapter 7, but here we must discuss the following more general issue.

We would like to represent differences explicitly as first-class objects so that we can reason about them during processing. For instance, in lexical choice, we might want to find the near-synonym that has the smallest difference to the input, or to a word in another language. But this idea is also necessary if we are to be faithful to Saussure’s view (see Hirst’s (1995) discussion on this point).

However, it would be impractical and inefficient to explicitly represent differences between the near-synonyms of a cluster. Not only can the set of differences be large—*n* near-synonyms have a maximum of $(n^2 - n)/2$ differences, pairwise—but each difference itself can be composed of a large set of distinctions (in denotation, attitude, and style). An average cluster of, say, five near-synonyms and six peripheral concepts might require $10 \times 6 = 60$ separate distinctions, not counting distinctions in style and attitude. Moreover, the same information would often be repeated in different distinctions. For example, to represent the distinctions in severity of criticism between the word *error* and four of its near-synonyms, we would represent that *error* has a level of severity greater than this word, less than this word, equal to that word, and so on. But this would be a very redundant and awkward way to represent the relative ordering, considering that we would have to represent all of the distinctions in this manner.

Besides that, lexicographers more often use positive terms in distinguishing near-synonyms; that is, differences tend to be described only implicitly in the entries. For example, in Gove’s entry for *mistake* (see figure 1.1, page 3), he says “*mistake* implies misconception, misunderstanding, a wrong but not always blameworthy judgment, or inadvertence”, which only implies a difference to *error* and *blunder*, whose definitions do not refer to these particular ideas. Of the 601 distinctions reported in table 3.8, page 37, 90% are stated in this manner.

When a lexicographer does explicitly differentiate two words it is usually phrased as a relative distinction. For example, Gove says *mistake* “expresses less severe criticism than *error*”, without stating the actual degree of severity involved. While such distinctions succinctly express the difference in meaning, they are difficult to use and apply in a system, because they provide only partial information. For instance, it would be difficult to find a word to match an input that asks for a word that expresses, say, low severity, if all we have represented are relative distinctions in severity.

Thus, it would seem more practical to represent all distinctions in positive terms, and to compute explicit differences from them only as needed. The one problem is in converting relative distinctions into positive terms, because we need a frame of reference in which to understand the distinctions. In the case of severity of criticism, for instance, we could define in the ontology different concepts for different degrees of severity, say, low, medium, and high severity, and then convert the assertion that *mistake* implies less severe criticism than *error* to two assertions that *mistake* implies low severity and *error* medium severity. However, it does not seem practical or even possible to define such absolute concepts, because there are many frames of reference: high severity for errors is different from high severity for words like *berate* and *scold*, and certainly different from the high severity of a hurricane.[5]

The solution is to treat the cluster as the frame of reference. Thus, we can specify in the *error* cluster that *slip*, *mistake*, *error*, and *blunder* express, respectively, low, low, medium, and high degrees of severity of criticism; but, crucially, these values can be taken to be meaningful only within the *error* cluster. That is, *mistake* expresses medium severity only with respect to, or relative to, its near-synonyms. This solution relies on the postulation that an unstated frame of reference does in fact underlie every relative distinction made by a lexicographer. But since each cluster represents a different frame of reference, we can assume very little about the relationships between different clusters, beyond the fact that they involve similar concepts. For instance, we can assume very little about the relationship between the degree of severity of criticism in the *error* cluster and the degree of severity of criticism associated with the words *berate* or *scold*, except that they all involve severity of criticism. Thus, one limitation of this solution is that it might be difficult to compare values on a given dimension between different clusters, and also between an input representation and a cluster. For now, we will ignore this limitation, but it does present an interesting direction for future research.

This same idea carries over to all types of distinctions made in a cluster, not just the overtly relative ones. That is, we take all distinctions to be relative within a cluster; the cluster establishes their local frame of reference. So, each word in a cluster is represented by a set of distinctions, which implicitly differentiate the word relative to its near-synonyms. (This approach, in effect, situates the word in a multi-dimensional space relative to its near-synonyms.) And, although differences between near-synonyms are represented implicitly, an explicit difference between two near-synonyms can be computed by comparing the sets of distinctions of the two words (see chapter 8).

5.4 Cross-linguistic clusters

Now, one of the main reasons for developing our theory is to support lexical choice in machine translation. For example, we would like a process that can choose the closest translation of, say, *error*, in another language. In the theory, we can in fact distinguish *error* from *bévue*, or any other alternative, just as we distinguish it from *blunder*. So, choosing the closest translation of *error* is

[5] Everyone knows that a small elephant is bigger than a large mouse: the meanings of *large* and *small* depend on the frame of reference established by the word that they modify, i.e., by the context. This phenomenon is the similar to the one described here.

reduced to comparing differences between *error* and each of its cross-linguistic near-synonyms (see section 3.6. In effect, 'language' is treated just like any other dimension of variation.

Now, there is a question as to whether clusters should be language-specific or cross-linguistic, that is, whether the near-synonyms must all be from the same language or whether they can come from many different languages. Figure 5.3 shows the former option, but this doesn't mean that the model cannot also account for the latter option. Clearly, each language might differentiate its synonyms in entirely different terms, requiring different peripheral concepts and stylistic dimensions to be defined. But this would not be a problem for a cross-linguistic clustered model, since we could simply represent all of the peripheral ideas in the same cluster, treating the words as if they were from the same language. What might be a problem, however, is if a cluster of near-synonyms in one language has a different, or slightly different, core denotation than a cluster in another language. Then we could not consider the words to be near-synonyms according to our definition. For now, we will assume that clusters in different languages share the same core denotation (as depicted in figure 5.3), which is reasonably tenable for 'close' languages. Hence, whether a cluster is language-specific or cross-linguistic is just a matter of implementation, and has more to do with the purpose to which the model will be used (i.e., text generation, machine translation, etc.). However, we will revisit to the theoretical problem in chapter 11.

5.5 Advantages of the theory

By introducing the subconceptual/stylistic level of semantic representation, we can now develop a new model of lexical knowledge that keeps the advantages of the conventional model—efficient paraphrasing, lexical choice (at a coarse grain), and mechanisms for reasoning—but overcomes its shortcomings concerning synonymy (which we listed in section 5.1 above).

Specifically, we can develop a model that uses a different, more expressive, formalism for representing fine-grained semantic knowledge (than is possible at the top level). Therefore, we can potentially solve the problem of inadequate coverage, because we can move beyond simple feature-based or truth-conditional formalisms, and so incorporate solutions to the problems of representing near-synonyms (implicit meanings, fuzzy distinctions, and so on), without adversely affecting the tractability or the efficiency of processing in the model.

And because such a model would avoid a complicated highly-cross-classified ontology, it also has engineering advantages over the conventional model. Lexicographers can concentrate their efforts where they have the expertise, on explicating the differences between near-synonyms and formalizing clusters of near-synonyms, while knowledge engineers can focus on the ontology.

A choice process that uses such a model would be efficient, because it could find the appropriate cluster or clusters just as easily as it could find words under a conventional model sans near-synonyms. Such a process would also be robust, ensuring that the right meaning (at a coarse grain) is lexicalized even if a 'poor' near-synonym is chosen in the end.

We develop two such models in chapters 6 and 7, below. Chapters 8 and 9 develop a sophisticated automatic lexical-choice process that uses the latter model. And chapter 10 demonstrates that the model and process do indeed hold up to these claims, which supports, if only indirectly, a theory of clustered near-synonyms.

5.6 Psycholinguistic predictions of the theory

Since our theory is relatively preliminary, our intent in this thesis is not to substantiate it per se, but rather to indirectly support it by using it as the basis for our two computational models. However, the theory does make empirically testable predictions.

For instance, the theory makes a prediction about the existence of core or essential meanings that have a fundamentally different status than peripheral meanings, and that the distinction is dependent on granularity of representation. To test this prediction, we could develop an experiment that attempts to determine if particular aspects of a word's meaning remain constant across different contexts and registers by eliciting responses from subjects. Much more research is required here.

The theory also makes a prediction concerning the ease with which people can compare and differentiate concepts. In a series of experiments, Markman and Gentner (Markman and Gentner, 1993; Gentner and Markman, 1994) showed that people can more easily articulate the differences between pairs of words that are similar in meaning (e.g., between *clock* and *watch*) than between pairs of dissimilar words (e.g., between *light bulb* and *cat*). (In one experiment, subjects were asked to list one difference each for as many pairs of words as they could during a restricted time interval. The assumption was that the subjects would do the 'easiest' pairs first.) Their explanation for this effect is based on their idea of *structural alignment*. Two conceptual structures are said to be *alignable* if they have many attributes in common, albeit with possibly different values for those attributes. Since greater similarity implies greater structural alignment, and since alignable differences are more salient in the comparison task—a separate empirical result—it is easier to differentiate pairs of similar words. Markman and Gentner then argue that a simple feature-based model of word meaning (i.e., a decompositional model) would make different predictions—under such a model, it would be no easier to list differences between similar pairs than between dissimilar pairs. So their results support a model of word meaning in which words are denoted by distinct but related conceptual structures—i.e., the ontological model.

However, Markman and Gentner did not consider near-synonyms in their experiments; their similar words were not close enough in meaning to be near-synonyms. If they had, what sort of results would they have seen? On the one hand, the ontological model predicts that it should be even easier to differentiate near-synonyms than merely similar words, since, in the model, near-synonyms denote distinct and structurally alignable concepts (which are even better aligned than the similar pairs). On the other hand, our clustered theory does not make this prediction, because structural alignment cannot be used to determine the differences between near-synonyms, which denote the same (coarse-grained) concept. The concepts are perfectly aligned, with differences encoded subconceptually. In fact, we might expect the task to be more difficult, because it seems more difficult for people to consciously access knowledge encoded at a lexical level (i.e., in lexical entries) than at a conceptual level (i.e., in conceptual structures).[6]

Therefore, if we were to repeat Markman and Gentner's experiment, this time including pairs of near-synonyms as well as pairs of similar and dissimilar words, our theory predicts that it would be easiest for subjects to list differences between the similar pairs. Hirst and Edmonds have begun to perform such experiments, but, so far, the results have been inconclusive. While subjects did seem to find it more difficult to list differences between near-synonyms (listing fewer differences on average), the proportion of alignable to non-alignable differences listed

[6] Even native speakers of a language often lack proficiency in the precise and effective use of near-synonyms. This is one reason why the market exists for reference books that explain the differences between near-synonyms.

was the same for near-synonyms as for similar words, indicating that subjects might in fact have been using some form of structural alignment in their comparisons.

Much more research is required in this area (see section 11.3.2), but since the focus of this thesis is on computational linguistics rather than psycholinguistics, we leave the work for another time and place. So, notwithstanding questions of psychological reality, in the next two chapters, we develop, respectively, a statistical model of differences in the usage of near-synonyms, and a knowledge-based model of differences in near-synonyms.

Chapter 6

A statistical model of the usage of near-synonyms

In this chapter, we discuss one very restricted form of the problem of lexical choice and near-synonyms.[1] Given a sentence with a 'gap' in it, and a set of candidate near-synonyms to fill the gap, how can we choose the candidate that is *most typical* or most natural in that sentence? By most typical, we mean the word that would most likely be chosen by an author under the same circumstances. Of course, a particular author might not make a typical choice (perhaps intentionally), but statistically over several authors, one (or more) of the synonyms would be more expected in the sentence. Moreover, the most typical word is not necessarily the best or most appropriate word. Presumably, a non-typical word would be marked in some way relative to the typical word, so a non-typical word might more accurately convey an author's desired meaning. So, while this lexical-choice problem is weaker than the general lexical-choice problem, it can be thought of as an important sub-problem, because it would enable a system to determine the effects of choosing a non-typical word in place of the typical word.

For example, in sentence 6.1, *error* would seem the most typical candidate (of *error, mistake,* and *oversight*). And in sentence 6.2, *mistake* would seem most typical. In both cases, these were the words chosen by the authors of the *Wall Street Journal*, from where these samples were taken.

6.1 The {error | mistake | oversight} was magnified when the Army failed to charge the standard percentage rate for packing and handling.

6.2 The incident is more than an embarrassment for officials in Perth; it may prove to be a costly {error | mistake | oversight}.

We will develop a solution to this problem that involves a new statistical approach to modelling the set of sentential (i.e., local) contexts that a word is (or can be) used in. We address the problem not by analyzing several authors' choices in the same contexts, since that would be impractical on a large scale, but by looking at what are, for practical purposes, the same author's choices made in different but similar contexts over a large corpus. The approach relies on a generalization of lexical co-occurrence that treats co-occurrence relations as though they were transitive.

The statistical model that we develop below is one formalization—albeit a pared down one—of the theory described in chapter 5. In the model, clusters of near-synonyms are assumed

[1] This chapter elaborates on previously published work (Edmonds, 1997); .

to share an unspecified conceptual meaning, which would be represented on the conceptual–semantic level. Then, on the subconceptual / stylistic–semantic level, we represent implicit differences between near-synonyms by means of statistical data about the differences in the choice of near-synonyms in different contexts.

6.1 Generalizing lexical co-occurrence

6.1.1 Statistical evidence-based models of context

Evidence-based models represent context as a set of features, say words, that are observed to co-occur with, and thereby predict, a word (Yarowsky, 1992; Church et al., 1994; Golding and Schabes, 1996; Ng and Lee, 1996; Karov and Edelman, 1998; Schütze, 1998). Thus, the meaning of a word, or rather, some aspect of its meaning and usage that we will refer to as its *contextual meaning*, is represented as a set of its co-occurring features, with associated significance scores, acquired from a large corpus. Differences between words, and near-synonyms, would be implicitly encoded as differences in their sets of features. For example, table 6.1 shows the top 20 words—here taken to be features—that co-occur with *error* and with *mistake* in a span of ±4 words in the 1989 *Wall Street Journal* corpus. The words are ranked according to *t*-score, and for comparison, mutual information scores are also shown. (The two measures will be defined below.) Notice that *error* differs from *mistake* in that it co-occurs with *margin/NN* and *rate/NN*, whereas *mistake* co-occurs with *big/JJ* and *stupid/JJ*. But also notice that both words co-occur with *was/VBD* (*error* at 1.74, *mistake* at 3.96), and *an/DT*, *a/DT* (2.74 and 4.88, respectively).

But can a particular instance of a context (i.e., a sentence with a gap) provide sufficient evidence to predict which of several near-synonyms is most typical in filling the gap? One suspects that it cannot, judging from sentences 6.1 and 6.2 above. Neither sentence contains sufficient words that appear in the sets of co-occurring words of *error* and *mistake* to differentiate the two.[2] Our experiments, reported below, bear this out.

Now, observe that even though a word might not co-occur significantly with another given word, it might nevertheless *predict* the use of that word, because the two words might be mutually related to a third word. That is, we can treat lexical co-occurrence as though it were moderately transitive, and thereby use a 'chain' of evidence. For example, in sentence 6.2, *costly* provides evidence for *mistake* because it co-occurs significantly with *taking*, which in turn co-occurs significantly with *mistake*, even though *costly* is not seen to co-occur significantly with *mistake*. The word *embarrassment* might also provide evidence through the chain *embarrassment* $\longrightarrow$ *political* $\longrightarrow$ *cutting* $\longrightarrow$ *mistake*.

So, by augmenting the set of features of a word with such *second-order* (and higher) co-occurrence relations, we stand to have greater predictive power, assuming that we assign less weight to them in accordance with their lower information content. And as the results of our experiment will show, this generalization of co-occurrence is necessary for differentiating the candidate words.

We can represent these relations in a *lexical co-occurrence network*, as in figure 6.1, that connects lexical items by just their first-order co-occurrence relations. Second- and higher-order relations are then implied by transitivity. In the figure, each relation is labelled with its significance (its *t*-score). The dashed edges represent higher-order relations, for which significance

[2] The research on word sense disambiguation shows that with this kind of contextual meaning-representation, it is difficult to distinguish even different senses of the same word, which presumably have meaning representations that differ more than those of near-synonyms.

$x =$ error / NN, $f(x) = 64$				
y	$f(y)$	$f(x,y)$	t	I
margin / NN	149	13	3.60	8.85
for / IN	26263	20	3.36	2.01
subgroups / NNS	9	8	2.83	12.20
direction / NN	161	8	2.82	8.04
an / DT	10139	11	2.74	2.52
example / NN	618	6	2.40	5.68
The / DT	22168	12	2.25	1.52
made / VBN	1063	4	1.90	4.32
rate / NN	1811	4	1.83	3.55
was / VBD	12667	7	1.74	1.55
human / JJ	216	3	1.71	6.20
have / VB	3631	4	1.66	2.54
some / DT	4040	4	1.62	2.39
no / DT	1871	3	1.53	3.08
because / IN	2912	3	1.41	2.45
Loantech / NP	5	2	1.41	11.05
drafting / NN	7	2	1.41	10.56
tactical / JJ	22	2	1.41	8.91
correct / VB	35	2	1.41	8.24
EDS / NP	43	2	1.41	7.94

$x =$ mistake / NN, $f(x) = 61$				
y	$f(y)$	$f(x,y)$	t	I
a / DT	60900	43	4.88	1.97
was / VBD	12667	20	3.96	3.13
it / PP	15141	14	3.01	2.36
made / VBD	688	9	2.96	6.18
big / JJ	1337	7	2.55	4.86
making / VBG	731	5	2.18	5.25
because / IN	2912	5	2.00	3.25
says / VBZ	6477	6	1.97	2.36
by / IN	14689	8	1.89	1.60
can / MD	2821	4	1.75	2.98
biggest / JJS	529	3	1.68	4.98
she / PP	1331	3	1.59	3.65
be / VB	12028	6	1.57	1.47
make / VB	1618	3	1.56	3.36
stupid / JJ	23	2	1.41	8.92
letting / VBG	49	2	1.41	7.82
cutting / VBG	156	2	1.39	6.15
anything / NN	315	2	1.37	5.14
think / VB	347	2	1.37	5.00
put / VB	370	2	1.37	4.91

Table 6.1: The top 20 words that co-occur with *error* and with *mistake* according to t-score in the 2,709,659-word part-of-speech tagged *Wall Street Journal* corpus. Each word is labelled with its P-O-S tag from a standard set of tags (e.g., NN refers to a singular noun, NNS to a plural noun, VBD to a verb in its past-tense form, JJ to an adjective, and so on). $f(x)$ and $f(y)$ are the frequencies of x and y, $f(x,y)$ is the frequency of y occurring within a span of ± 4 words around x. t and I are different measures of significance.

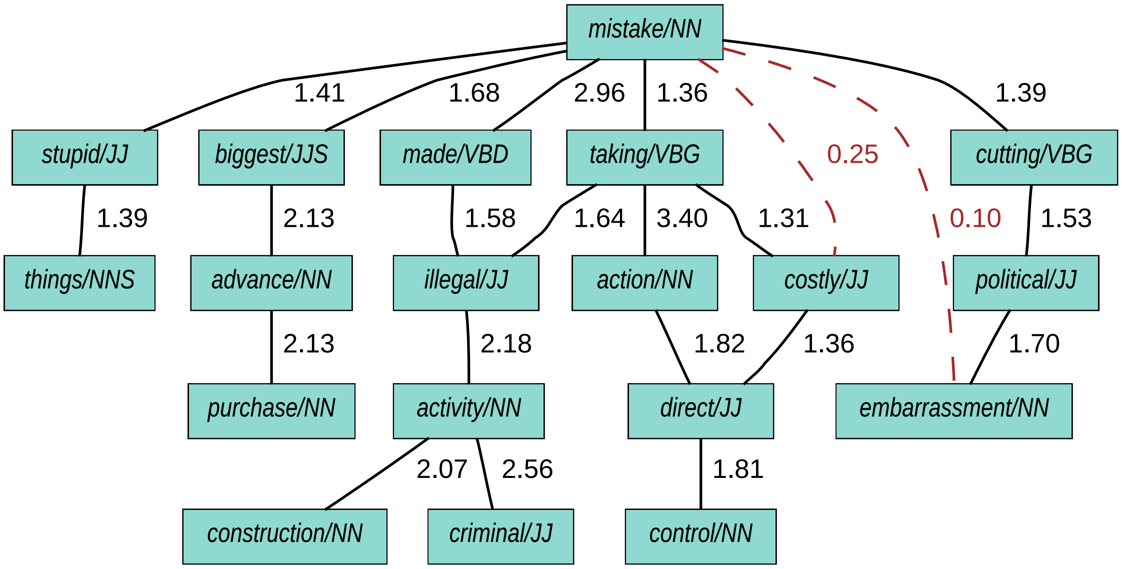


Figure 6.1: A fragment of the lexical co-occurrence network for *mistake*. The dashed lines are implicit higher-order relations in the network.

scores have to be computed from the intermediate relations, as described below. Figure 6.1 is only an illustrative fragment of the network for *mistake*. The complete network, comprising 6,482 nodes and 19,278 edges (allowing up to third-order relations, and subject to the parameters discussed below), represents the contextual meaning of *mistake* as used in the *Wall Street Journal* corpus.

6.1.2 Lexical co-occurrence networks

We build a lexical co-occurrence network as follows: Given a root word, connect it to all the words that significantly co-occur with it in the training corpus; then, recursively connect these words to their significant co-occurring words up to some specified depth. Several parameters are involved in the process, including what counts as a ‘word’, the criteria for admitting a word to the network, the size of the span of words considered as local context, and the method of computing significance scores for both first-order and higher-order relations.

What counts as a word? We used the part-of-speech tagged 1989 *Wall Street Journal* as our training corpus. It has $N = 2{,}709{,}659$ tokens of 95,981 different kinds. A token is any word, number, or symbol tagged for part-of-speech. So, for us, a ‘word’ is any token not on our stop list, which contains all numbers, symbols, and proper nouns (a total of 34,509 tokens). The stop list also contains any token with a raw frequency greater than $F = 800$. The value of F was determined over several trial runs of the experiment described in section 6.2 below, in which F was varied while holding the other parameters constant. Setting $F = 800$ gave us the best results , and added 328 high-frequency words to the stop list. Thus, our lexicon consists of 61,144 words. Note that *mistake/NN* and *mistake/VB* are distinct words, as are *settle/VB* and *settled/VBD*, but since no sense disambiguation is done, *job/NN* represents all of the senses of the noun *job*.

Admission criteria Only the words that co-occur most significantly with a particular word should be admitted to the network, but we had to determine a suitable threshold. The threshold has consequences for the efficiency and accuracy of the lexical choice procedure: if too many words are admitted, then the network can become too large to process efficiently, and the co-occurrence relations may be too weak or insignificant to be of use or may even add too much 'noise' to the procedure; if admission is overly constrained, then the network will not contain enough detail to differentiate the root word from its near-synonyms.

In addition, we had to decide which measure of significance to use. The two most well-known measures are the mutual information statistic (equation 6.1) and the t-test (equation 6.2).

$$I(x,y) = \log_2 \frac{\Pr(x,y)}{\Pr(x)\Pr(y)s} = \log_2 \frac{\frac{f(x,y)}{N}}{\frac{f(x)}{N}\frac{f(y)}{N}s} = \log_2 \left(\frac{f(x,y)}{f(x)f(y)} \frac{N}{s} \right) \tag{6.1}$$

$$t(x,y) \approx \frac{\Pr(x,y) - \Pr(x)\Pr(y)s}{\sqrt{\Pr(x,y)/(N-1)}} \approx \frac{f(x,y) - f(x)f(y)\frac{s}{N}}{\sqrt{f(x,y)}} \tag{6.2}$$

In these equations, the probability $\Pr(x)$ of a word x occurring in the corpus is computed as $f(x)/N$, where $f(x)$ is the observed frequency of x in the corpus of N words. The joint probability $\Pr(x,y)$ of y occurring within a span of s words centered on x is computed in the same way. (Note: the chance probability of y occurring in a span around x is $\Pr(x)\Pr(y)s$.)

Through trial experiments, we determined that using either measure on its own was inadequate for building a useful network. The reason why is evident from table 6.1. Each measure favours a different kind of co-occurrence relation: t-scores favour grammatical collocations, or collocations with relatively frequent words (and high joint frequencies); I-scores instead favour collocations with less frequent words. Thus we used a combination of the two measures (Church et al., 1991). After more trial runs of the experiment described below, it was determined that thresholds of 2.0 and 3.0, for t-score and I-score, respectively, gave the best performance. Thus, any word with a t-score greater than 2.0 and an I-score greater than 3.0 is admitted to the network.

Span size We can also vary the size, s, of the span of context. As equations (6.1) and (6.2) show, span size affects the significance measures: a co-occurrence that is significant in a narrow span might not be in a wider span. What span size is optimal for this task, if one exists, is an interesting question. For instance, Gale, Church and Yarowsky (1992) found that a span of ±50 words was useful in broad topic classification (an aspect of word sense disambiguation), but it is not clear whether the same span is useful in all tasks. We address this issue empirically in our experiment in section 6.2, below.

Calculation of significance Every relation between words w_0 and w_d in the network must be assigned a *significance score*, $sig(w_0, w_d)$, in which d is the order of the relation. For first-order relations ($d = 1$) we can use one of the significance measures described above, or a linear combination of the two. Through further trial experiments we determined that using just t-scores worked better that using just I-scores, and also better than a linear combination. Thus,

$$sig(w_0, w_1) = t(w_0, w_1) \tag{6.3}$$

We compute significance scores for second- and higher-order relations using the t-scores of the relations on the shortest path between the related words. Formally, for $d > 1$, given an arbitrary shortest path $P(w_0, w_d) = (w_0 \rightarrow w_1 \rightarrow \cdots \rightarrow w_d)$ between the words w_0 and w_d, the significance score is

$$sig(w_0, w_d) = \frac{1}{d^3} \sum_{w_i \in P(w_1, w_d)} \frac{t(w_{i-1}, w_i)}{i} \tag{6.4}$$

So, for example, in the network of figure 6.1,

$$sig(mistake, costly) = [t(mistake, taking) + \tfrac{1}{2}t(taking, costly)]/8 = (1.36 + 1.31/2)/8 = 0.25$$

Equation (6.4) subsumes equation (6.3), and it ensures that significance is inversely proportional to the order of the relation by dividing by a factor of d^3. It also gives progressively less weight to relations further along the path from w_0. Many other formulas are possible. Again, we experimented with different formulas that had different exponents on d and different weightings within the sum, but we found this formula worked the best. Nevertheless, this formula is only preliminary, so we make no theoretical claims at this point.

6.2 Experiment

6.2.1 Method: Choosing the most typical word

The amount of evidence that a given sentence provides for choosing a candidate word is the sum of the significance scores of each co-occurrence of the candidate with a word in the sentence. So, given a gap in a sentence S, we find the candidate c for the gap that maximizes

$$M(c, S) = \sum_{w \in S} sig(c, w) \tag{6.5}$$

For example, given S as sentence 6.2, above, and the complete networks for *error, mistake,* and *oversight*, $M(error, S) = 0.568$, $M(mistake, S) = 0.634$, and $M(oversight, S) = 0.339$; thus *mistake* is chosen as most typical in S.

We implemented formula (6.5) in our lexical choice program, which takes as input a sentence with a gap, a cluster of near-synonyms, and a few parameters. It outputs the most typical near-synonym, that is, the one that maximizes formula 6.5. If two or more of the candidates maximize the formula, then the program does not make a choice. However, this usually happens only when there is no evidence for selecting any of the near-synonyms (i.e., when $M(c, S) = 0$ for all candidates c).

To evaluate the program, we selected seven clusters of near-synonyms, shown in table 6.2, in which each near-synonym occurs with similar frequencies to its mates. We also tried to select clusters having low polysemy in the corpus in order to reduce the confounding effects of lexical ambiguity, which we discuss below.

For each cluster, we collected all sentences from the previously-unseen 1987 *Wall Street Journal* (part-of-speech-tagged) that contained any of the members of the cluster, ignoring word sense. We replaced each occurrence by a 'gap' that the program then had to fill. This formed our collection of test sentences (or contexts). We compared the 'correctness' of the choices made by our program to the baseline of always choosing the most frequent synonym according to the training corpus.

But what are the 'correct' responses? Ideally, they should be chosen by a credible human informant. But regrettably, we are unaware of a study that asks informants to judge typical

Cluster	P-O-S	Near-synonyms
DIFF	JJ	difficult (352), hard (348), tough (230)
ERROR	NN	error (64), mistake (61), oversight (37)
JOB	NN	job (418), task (123), duty (48)
RESPTY	NN	responsibility (142), commitment (122), obligation (96), burden (81)
MATRL	NN	material (177), stuff (79), substance (45)
GIVE	VB	give (624), provide (501), offer (302)
SETTLE	VB	settle (126), resolve (79)

Table 6.2: The clusters of synonyms for our experiment. Frequency in the training corpus is indicated in brackets.

Cluster		DIFF	ERROR	JOB	RESPTY	MATRL	GIVE	SETTLE
Sentences		6,665	1,030	5,402	3,138	1,828	10,204	1,568
Baseline		40.1	33.5	74.2	36.6	62.8	45.7	62.2
Narrow	1	31.3	18.7	34.5	27.7	28.8	33.2	41.3
	2	47.2	44.5	66.2	43.9	61.9[a]	48.1	62.8[a]
	3	**47.9**	**48.9**	**68.9**	44.3	**64.6**[a]	**48.6**	**65.9**
Medium	1	24.0	25.0	26.4	29.3	28.8	20.6	44.2
	2	42.5	47.1	55.3	**45.3**	61.5[a]	44.3	63.6[a]
	3	42.5	47.0	53.6	—	—	—	—
Wide	1	9.2	20.6	17.5	20.7	21.2	4.1	26.5
	2	39.9[a]	46.2	47.1	43.2	52.7	37.7	58.6
	3	—	—	—	—	—	—	—

[a] Difference from baseline not significant.

Table 6.3: Percentage accuracy (i.e., recall) of several different runs of the lexical choice process on each of the seven synonym clusters. The best score for each cluster is in boldface. All differences from baseline are significant at the 0.05 level according to Pearson's χ^2 test, unless indicated.

usage, and we are not in a position to undertake a study of our own, so we will turn instead to a less ideal source: the authors of the *Wall Street Journal*. The problem is, of course, that authors aren't always typical. As we said, authors strive to make the best or most appropriate choices, which are certainly not always the same as the most typical choices. So, if an author chose one particular word for a context in which another synonym was seen more often, the word would not be the typical choice. Thus, we cannot expect perfect accuracy in this evaluation. However, one might expect perfect accuracy to be attainable in the limit of the size of the training corpus.

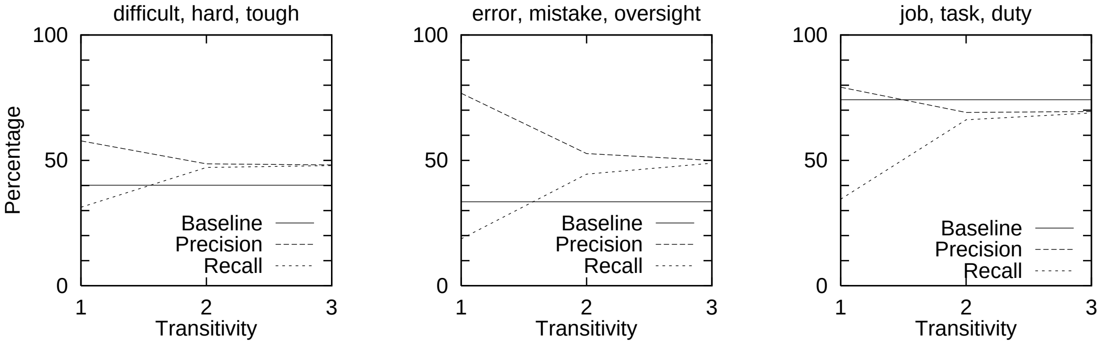


Figure 6.2: Recall and precision of our lexical choice procedure compared to baseline performance varying transitivity on three clusters of synonyms.

6.2.2 Results and Evaluation

Table 6.3 shows the results for the seven synonym clusters over several runs each while varying two parameters. As discussed above, we varied the span size: either narrow (±4 words), medium (± 10 words), or wide (± 50 words). (The medium setting corresponds to the average sentence length of 20 words in the corpus.) The second parameter that we varied was the maximum transitivity (or order) of co-occurrence relation allowed: either 1, 2, or 3. (A setting of 1 coincides with using only regular (first-order) co-occurrence relations.) In all, this would have given $3 \times 3 = 9$ separate runs per synonym cluster, except we omitted the combination of the wide span with transitivity 3 because of prohibitive time and space requirements.

The results show that at least second-order co-occurrences are necessary to achieve better-than-baseline accuracy in this task; regular co-occurrence relations are insufficient. Third-order relations add still more accuracy, but the law of diminishing returns is at work. This justifies our assumption that we need more than the surrounding context to build adequate contextual representations. It also shows that co-occurrence relations may be transitive, at least to some degree, which is an intriguing possibility.

Also, the narrow span gives consistently higher accuracy than the other sizes. This lends support to the idea that differences between near-synonyms often involve differences in short-distance collocations with neighboring words, e.g., *face the task*.

Even though our program is on average only just better than the baseline, it is qualitatively different than the baseline procedure. It is more conservative, because it only makes a choice when it has sufficient evidence to differentiate the candidates. The baseline, on the other hand, always makes a 'choice' and always 'chooses' the same word. To highlight the difference, we can use the *recall* and *precision* statistics commonly used in the evaluation of information retrieval systems. Here, recall is the percentage of all test sentences for which a correct choice is made (recall is what is shown in table 6.3 above); precision is the percentage of choices made that are correct. Figure 6.2 shows the recall and precision of our program on three clusters of synonyms while varying the maximum allowable transitivity. Since the baseline always chooses a word, its recall equals its precision. Notice that our program trades high recall for high precision when only first-order co-occurrence relations are used; so, while the program is often unable to make a choice, when it does choose, it is much more likely to make the correct choice than the baseline. Also notice that recall and precision converge as transitivity is increased, indicating that the procedure sacrifices precision in order to make more choices.

6.2.3 A more comprehensive experiment

In order to make sure that our results were not dependent on the particular clusters of near-synonyms that we chose, we ran our experiment on all WordNet 1.5 synsets. We first selected the synsets for which at least two members occurred in the 1989 *Wall Street Journal* (the training corpus) and for which the members' average frequency of occurrence was greater than 50. Since the corpus is part-of-speech tagged, we inflected each word (depending on its syntactic category) and added the appropriate P-O-S tag. For verbs, we used the infinitive, present-tense (first and third person), and past-tense forms; for nouns, singular and plural forms; and for adjectives no inflection was required. This left us with 2,103 inflected synsets, with an average of about 3,200 test sentences (from the 1987 *Wall Street Journal*) per synset.

For each synset, we ran the same experiment as above, except that we fixed the span size to ±4 words and the transitivity to 2. The total processing time was 87 hours (on a Sun SPARCstation 5), or about 2.5 minutes per synset.

The average baseline, recall, and precision scores over all 2,103 synsets are the following:

Baseline	73.3%
Recall	67.9%
Precision	74.5%

The differences from the baseline for both recall and precision are significant at the 0.05 confidence level according to a standard *t*-test. Thus, our algorithm performed almost as well as the baseline in recall, and slightly exceeded it in precision.

6.3 Analysis

There are two reasons why the approach doesn't do as well as an automatic approach ought to. First, lexical ambiguity is a major problem. Our results are best for synonym clusters that have a low polysemy in the corpus. For instance, the program performed very well on the *error* synonyms, which probably have the lowest polysemy of all seven clusters. *Error*, *mistake*, and *oversight* are almost always used in the same sense in the training corpus, though *oversight* is sometimes used as 'supervision'. On the other hand, *job, task,* and *duty* are quite ambiguous in the corpus—*job*, especially, is more often used as 'occupation' than in our desired sense. Consequently, the results are quite poor. The reason is that during the construction of the network, wrong senses add irrelevant words to the network, which when taken together can confound the lexical choice process. For instance, in example 6.3 below, *safety* is a red herring: it is used as evidence for choosing *job* (because *job safety* is a frequent collocation), but this is the wrong sense of *job*.

6.3 The team's most urgent {task | job | duty} was to learn whether Chernobyl would suggest any safety flaws at KWU-designed plants.

Unfortunately, until a large sense-tagged corpus is available, there is little we can do to remedy this problem.[3]

[3] SemCor, a 676,000-word subset of the *Brown corpus* in which about 235,000 words are each tagged with the WordNet 1.6 synset to which they belong, is not large enough to give us significant results. For example, if we treat the entire corpus as the training corpus, then only 102 synsets meet the criteria of section 6.2.3, but we would have no sample sentences left to test the program on.

The second reason is that our representation of contextual meaning is inadequate. The approach was intended to be simple and easy to implement in order to see how far a basic approach could take us, and as the results show, it is insufficient. The next step should be to extend the model of context and incorporate different kinds of features, including aspects of morphology, collocational structure, syntactic structure, and semantic classes of words (Yarowsky, 1994; Grefenstette, 1994; Ng and Lee, 1996; Golding and Schabes, 1996; Karov and Edelman, 1998; Schütze, 1998). Also, we did not exploit the full power of the lexical co-occurrence networks and their transitivity relations. The networks have a very complex structure, which we did not even begin to investigate.

In conclusion, we investigated the feasibility of using a basic statistical model of context to represent differences in the usage of near-synonyms. The minor success of our generalization of lexical co-occurrence is encouraging—especially the notion of transitive co-occurrence relations—however, a significant amount of research and empirical investigation remains to be done. A more sophisticated model of context is required, but this is hindered by the problems caused by lexical ambiguity, a currently open problem.

Furthermore, a statistical model can only represent typicality of use in context, which is insufficient for goal-directed NLG and MT, unless of course the goal is to be typical. For true goal-directed lexical choice, we need a knowledge-based model, but that does not mean it must supplant a statistical model, for the two types of model are complementary. In the next chapter, we will describe a knowledge-based model of lexical meaning.

Chapter 7

A clustered model of lexical knowledge

This chapter develops a computational model for representing the fine-grained meanings of near-synonyms called the *clustered model of lexical knowledge.* The model is a formalization of the theory described in chapter 5. In the model, clusters of near-synonyms, rather than individual words, are linked to an ontology. Within a cluster, the near-synonyms are implicitly differentiated in terms of peripheral concepts, style, and expressed attitude. In fact, a cluster acts as a *formal usage note* (DiMarco, Hirst and Stede's (1993) term) that explicitly specifies how a system can both lexicalize fine-grained meanings and recover lexical nuances from text.

The model takes the conventional ontological model as a starting point; however the new model is better suited to representing near-synonyms because it

- cuts off the ontology at a coarse grain, which alleviates an awkward proliferation of language-dependent concepts on the fringes of the ontology,

- moves away from hyponymy as the most important inter-word relation, treating all relations (of difference) equally,

- enables the representation of features and concepts that are not necessary-and-sufficient, which allows the representation of indirect meanings and context-dependent meanings,

- enables the differentiation of words in terms of complete concepts and fuzzy concepts, rather than features, so it can account for the possibly complex relationships between the aspects of the meanings of a group of near-synonyms, and

- enables the differentiation of words in terms of expressed attitude and style.

The model is designed to be used in a natural language processing system that either analyzes text to construct a representation of its meaning, or generates text from a meaning representation. While processing issues will be covered in chapters 8 and 9, we will make some references to processing as needed in this chapter. Note also, that we focus on applications of the model in generation and lexical choice, or both.

Also, while the model currently does not account for how the context might influence the various processing tasks, it doesn't inherently exclude an account either. Since the model is designed to be used in a natural language processing system, the effects of context can be integrated into it as the context-dependent nature of words becomes better understood.

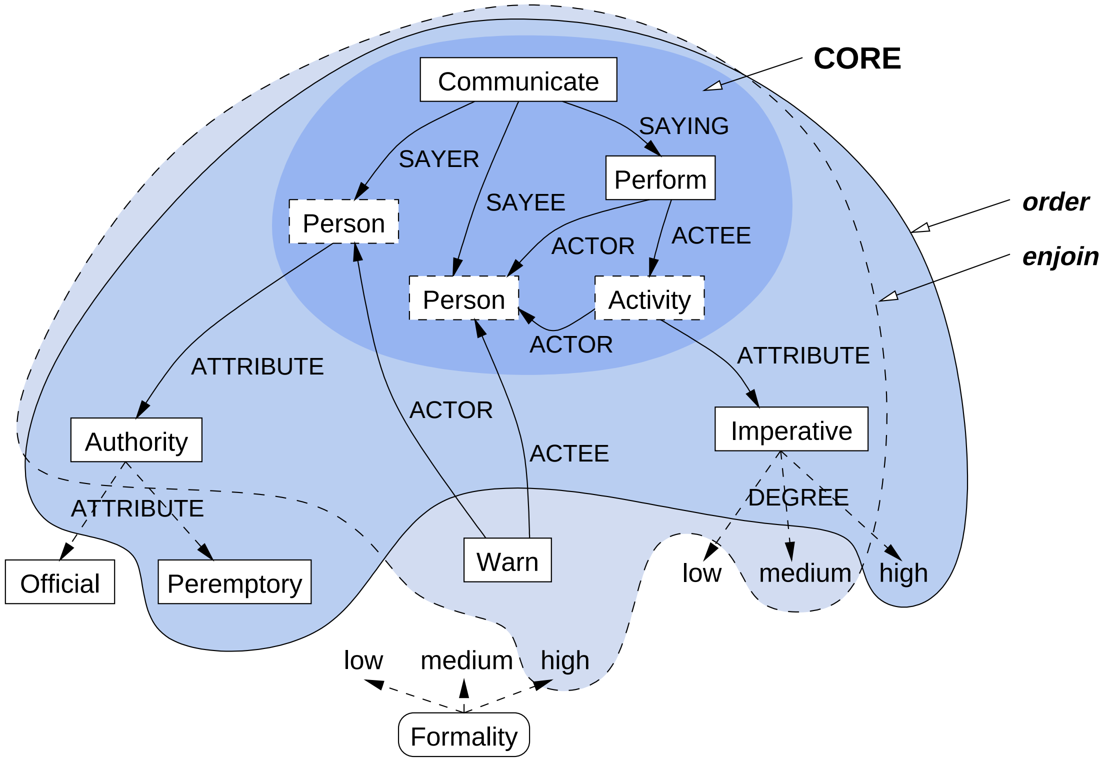


Figure 7.1: The core denotation and peripheral concepts of the cluster of *order* verbs. The two lightly shaded regions show the concepts that can be conveyed by the words *order* and *enjoin* in relation to each other.

7.1 Overview of the clustered model

The clustered model of lexical knowledge has the following components:

- A language-neutral ontology, the concepts of which are used in constructing representations of word meaning (section 7.2).

- Language-specific clusters of near-synonyms that explicitly differentiate the near-synonyms (section 7.3).

- A set of lexical entries, one for each word sense, that characterize how each word can be combined with other words at a syntactic–semantic level (section 7.4).

The first and third components provide the necessary support to represent the fine-grained meanings of near-synonyms; however, the heart of the model is in the second component, the clusters themselves. As an overview of the model, consider the following two examples, respectively, of the cluster representations for the *order* verbs and the *error* nouns.

Figure 7.1 depicts part of the representation of the cluster of *order* verbs. It shows the core denotation (the darkly shaded region), three peripheral concepts (the concepts in the lightly shaded regions linked to the core concept by solid arrows), and the stylistic dimension of 'formality'. The core denotation and peripheral concepts together form a directed graph of con-

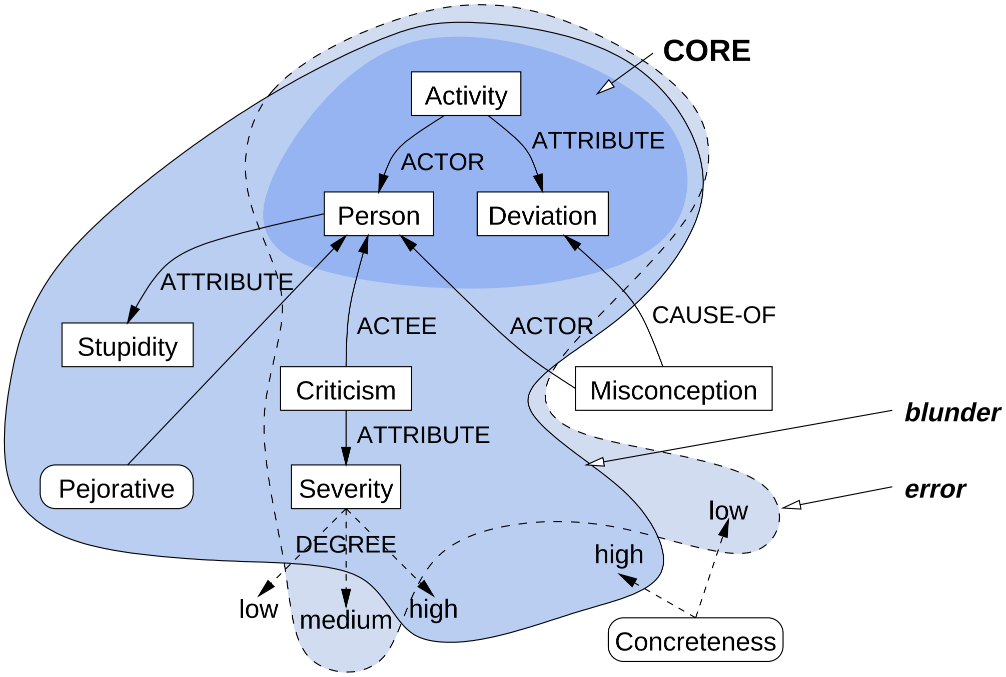


Figure 7.2: The core denotation and some of the peripheral concepts of the cluster of *error* nouns. The two lightly shaded regions show the concepts (and attitudes and styles) that can be conveyed by the words *error* and *blunder* in relation to each.

cepts linked by relations; the individual concepts and relations are defined in the ontology. (Note that concepts are depicted as regular rectangles, whereas stylistic dimensions and attitudes are depicted as rounded rectangles.) In this cluster, the core represents a communication by a person (the sayer) to another person (the sayee) of an activity that the sayee must do. The peripheral concepts represent that a near-synonym of the cluster can potentially express the authority of the sayer (with possible values of `Official` or `Peremptory`), a warning to the sayee, and/or the imperativeness of the activity (with possible values of `low`, `medium`, or `high`). But while all of the near-synonyms in the cluster will convey the concepts in the core denotation, the peripheral concepts that will be conveyed depend on each near-synonym. This is depicted by the two lightly shaded regions, which each contain the concepts (and styles) conveyed by their associated near-synonyms. Thus, *order* can convey a high degree of imperativeness compared to the medium degree that *enjoin* can convey; *order* can also convey peremptory authority, but *enjoin* can convey mere authority, without specifying the type of authority. Finally, *enjoin* can convey a warning to the sayee. (Note that the indirectness of expression of each of peripheral concepts by each of the near-synonyms is not shown in this diagram.)

Figure 7.2 depicts the cluster of *error* nouns, including three of its peripheral concepts (stupidity of the actor of the error, severity of criticism, and misconception as cause), the stylistic dimensions of ‘concreteness’, and also a pejorative attitude towards the actor. As in figure 7.1, the darkly shaded region contains the core denotation of the cluster, and the two lightly shaded regions contain the sets of concepts, styles, and attitudes, that can be conveyed by *error* and *blunder*, respectively. However, the attitude connects to the concept `Person`, because all attitudes must be directed towards an entity of the situation to be expressed. Stylistic dimensions, on the other hand, are completely separate from the graph of concepts.

7.2 The ontology in the clustered model

7.2.1 General considerations

A comprehensive and well-founded ontology is crucial to a knowledge-based model of lexical meaning, because all words must ultimately be defined in terms of concepts in the ontology. But should we build a model that relies on such an ontology? Ontologies have been successfully used in other projects (as we saw in sections 4.1.1 and 4.2.5), but we must still address a few concerns specific to our objectives.

One underlying goal of this research is to develop a general multilingual lexical resource—a computational thesaurus—that can be used in a variety of applications and domains. So, the resource should be as application- and domain-independent as possible. While this is certainly a desirable objective, it is not clear whether it is possible to develop such a resource without the need for extensive reorganization or rewriting of the ontology and of the lexical entries for each new application.

For one thing, the clustered model makes the assumption that an ontology diverges from language-neutrality into language-dependence only at the points where fine-grained specializations of concepts might occur. That is, it assumes that the overall structure of the ontology is not dependent on any particular language.[1] It might not be possible to build such an ontology. It is unlikely that one 'true' ontology exists, even within a single culture, let alone different language groups, because there are, of course, many valid ways of conceptualizing the world. This has been a main criticism of the ontological approach.

Research so far indicates that a pragmatic stance might be best. That is, rather than concern oneself with questions of psychological plausibility and encyclopedic knowledge, it seems better to represent only the knowledge that helps one's system perform better. For example, in the Mikrokosmos project (Mahesh and Nirenburg, 1995) only the type of knowledge that is found to be required for a specific application, in this case, machine translation of a certain kind of document, is represented. Domain-independence is striven for, but is not always possible. This stance is at odds with the desire to build a truly general resource.

It is possible to develop many different, yet comprehensive and well-founded, ontologies. However, we can expect that most ontologies will be similar enough that mapping from one to another can be done reasonably easily (see work on EuroWordNet for mapping between the WordNets of different languages (Vossen, Díez-Orzas and Peters, 1997; Vossen, Peters and Gonzalos, 1999)). And, as discussed in section 4.1.1, a standard ontology for natural language processing is receiving sporadic attention.

Therefore, while we believe that it is a viable goal to develop a general lexical resource with a general ontology—lengthy, perpetual even, though the process may be—it will be more advantageous to take the pragmatic stance, for now. We will thus develop an ontology whose specific purpose is to enable the demonstration of the capabilities of the clustered model. In any case, the focus of this research is on the framework for representing lexical meaning, not on issues of ontological representation, reasoning, or knowledge acquisition. The complete 'pro-

[1] For a non-multilingual application, such as an NLG system, language-independence of concepts might be irrelevant. This does not hurt the model itself, but it does require that some other criteria for membership in the ontology be determined. This issue requires further research, but is outside the scope of this thesis.

duction' ontology would and should be built by lexicographers and knowledge engineers.

7.2.2 Technical requirements of the ontology

We must choose a knowledge representation language (and system) that is sufficiently expressive and computationally tractable. Our minimal requirements for this system, the reasons for which will become clear throughout this chapter, are as follows:

A. Representational

1. Can define concepts in terms of other concepts.
2. Supports inheritance.
3. Can define arbitrary relations between concepts (e.g., agent, attribute, etc.), and constraints on the related concepts.
4. Can instantiate concepts.
5. Can define graph structures of concepts, not just trees.
6. Can define non-necessary components of meaning, or varying levels of necessity.
7. Can define fuzzy concepts.

B. Computational

1. Can make assertions about instances of concepts.
2. Can compute subsumption relationships.
3. Can enforce well-formedness constraints on instances.
4. Queries (e.g., to find superconcepts, or relations between concepts or instances) are computationally tractable.
5. Can compute quantitative similarity of concepts.

While most of these requirements are fairly standard and supported by frame-based knowledge representation languages (i.e., by description logics), requirements A5–A7 and B5 are not generally supported. In fact, these four require at least first-order logic, which could render queries in our system too computationally expensive. (Loom (MacGregor and Bates, 1987), for instance, allows one to represent more complex structures, e.g., for requirement A5, but this disables its subsumption mechanism.) The typical solution to such matters, taken, for instance, in the Mikrokosmos project (Mahesh and Nirenburg, 1995) and by Stede (1996), is to use a frame-based language in the ontology, and to push the more complex representational issues into the lexicon, where a more expressive language can be used without as much loss of tractability.

Also, from a practical standpoint, we need to integrate our lexical choice algorithm into an existing sentence generation system. Since we are building our system on top of Stede's MOOSE system (Stede, 1996), which uses the Loom knowledge representation system, we will also use Loom.

7.2.3 Our ontology

As indicated above, we built a special purpose ontology that defines only the concepts and relations that are necessary to demonstrate our model's capabilities. It is currently quite small with about 100 concepts and 40 relations. Figure 7.3 shows a sample of the concepts defined

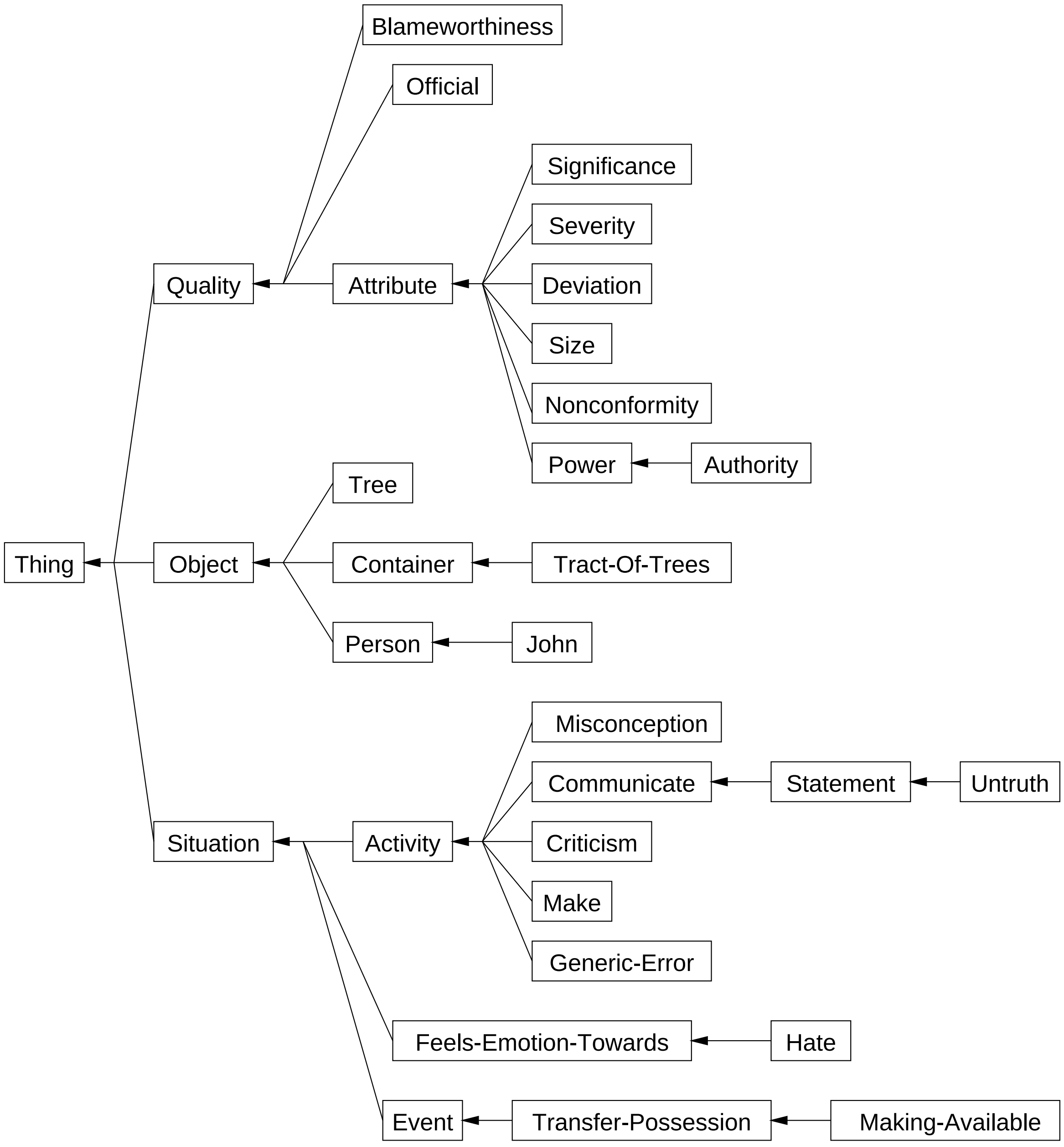


Figure 7.3: A portion of the concepts and their subsumption relationships in our ontology.

```
(defconcept MyThing)

(defconcept Quality :is-primitive MyThing)
(defrelation QUALITY :domain MyThing :range Quality)

(defconcept Object :is-primitive MyThing)

(defconcept Situation :is-primitive MyThing)
```

Listing 7.1: Loom definitions of the top level concepts.

in our ontology with their subsumption relationships. It does not show other relationships between concepts, nor does it show constraints on those relationships, but we'll discuss those below. We make no claims about the 'correctness' of the ontology; however we based it on other ontologies including Mikrokosmos, Stede's ontology, WordNet, and CYC. The top level of the ontology reflects the typical distinctions made in ontologies of qualities, objects, and situations. Listing 7.1 shows the Loom definitions of the top level categories. (We follow the convention of capitalizing concepts and of using uppercase for relations.)

Object The `Object` hierarchy is currently very basic (see listing 7.2). A `Person` is defined simply as a type of `Object`. A `Tract-Of-Trees` is a `Container` that contains some number of `Trees`.

Quality Qualities are the properties and attributes that objects, situations, and other qualities have. They are most often lexicalized in English by adjectives, but need not be. In the ontology (see listing 7.3), an `Attribute` is a `Quality` that is `Scalable`, which means it can have a `DEGREE`. Thus, we define simple binary qualities, such as `Blameworthiness`, which an object or situation either has or has not, and continuous dimensions, such as `Severity` and `Size`. Since any quality can have a quality related to it, we can also define the multi-value dimensions such as `Authority`, which can have attributes of `Official` or `Peremptory`.

Situation Situations are the most complex concepts because they must relate various participants to one another. We used the upper ontology that Stede developed for situations. Stede, after careful research, defined three subtypes of situation: states, events, and activities. Listing 7.4 shows only `Activity` and its subtypes. An `Activity` is a `Situation` in which something is happening (rather than being static). Thus, an `Activity` can have an `ACTOR`, an `ACTEE`, and a `CAUSER` (who might be the same entity as the `ACTOR`). For instance, a `Generic-Error` is an `Activity` that is a `Deviation`, and whose `ACTOR` is a `Person`. `Communicate` defines an `Activity` that involves a `SAYER`, a `SAYEE`, and a `SAYING`. A `Statement` is a subtype of `Communicate`, and an `Untruth` is a statement that has a `Nonconformity`. (Alternatively, we could have defined statements and errors as types of objects, but as we said, we are not concerned about such issues in ontological modelling, so long as our ontology is based on realistic assumptions.)

The complete ontology is given in appendix A.

```
(defconcept Person :is-primitive Object)
(defconcept John :is-primitive Person)

(defconcept Container :is-primitive Object)
(defrelation CONTAINS :domain Container :range Object)
(defconcept Tract-Of-Land :is-primitive Container)
(defconcept Tract-Of-Trees :is (:and Tract-Of-Land (:some CONTAINS Tree)))
```

Listing 7.2: Loom definitions of some objects.

```
(defconcept Official :is-primitive Quality)
(defconcept Peremptory :is-primitive Quality)
(defconcept Blameworthiness :is-primitive Quality)

(defconcept Scalable :is-primitive (:at-most 1 DEGREE))
(defconcept ScaleValue :is (:through 0.0 1.0))
(defrelation DEGREE :domain Scalable :range ScaleValue
                    :characteristics :single-valued)

(defconcept Attribute :is-primitive (:and Quality Scalable))
(defrelation ATTRIBUTE :is-primitive QUALITY
                       :domain MyThing :range Attribute)
(defrelation ATTRIBUTE-OF :is (:inverse ATTRIBUTE))

(defconcept Deviation :is-primitive Attribute)
(defconcept Nonconformity :is-primitive Attribute)
(defconcept Severity :is-primitive Attribute)
(defconcept Significance :is-primitive Attribute)
(defconcept Size :is-primitive Attribute)
(defconcept Power :is-primitive Attribute)
(defconcept Authority :is
                      (:and Power
                            (:at-most 1 ATTRIBUTE)
                            (:all ATTRIBUTE (:or Peremptory Official))))
```

Listing 7.3: Loom definitions of some qualities and attributes.

```
(defconcept Activity :is-primitive (:and Situation
                                         (:at-most 1 ACTOR)
                                         (:at-most 1 ACTEE)
                                         (:at-most 1 ACT-CAUSER)))
(defrelation ACTOR :domain Activity :range MyThing)
(defrelation ACTEE :domain Activity :range MyThing)
(defrelation ACTOR-OF :is (:inverse ACTOR))
(defrelation ACTEE-OF :is (:inverse ACTEE))
(defrelation ACT-CAUSER :domain Activity :range Person)

(defconcept Criticism :is-primitive Activity)
(defconcept Generic-Error :is (:and Activity
                                    (:all ACTOR Person)
                                    (:some ATTRIBUTE Deviation)))
(defconcept Make :is-primitive Activity)

(defconcept Communicate :is (:and Activity
                                  (:exactly 1 SAYER)
                                  (:exactly 1 SAYEE)
                                  (:exactly 1 SAYING)))
(defrelation SAYER :is (:and ACTOR (:range Person)))
(defrelation SAYEE :is (:and ACTEE (:range Person)))
(defrelation SAYING :domain Communicate :range MyThing)

(defconcept Statement :is-primitive Communicate)
(defconcept Untruth :is (:and Statement
                              (:some ATTRIBUTE Nonconformity)))
```

Listing 7.4: Loom definitions of some situations.

7.3 Clusters of near-synonyms

As described above, a cluster is intended to be a formal usage note that mirrors a dictionary usage note. A cluster has the following fields:

Syns A list of the near-synonyms in the cluster.

Core The core denotation, or essential shared meaning of the near-synonyms in the cluster. It is represented as a configuration of concepts.

Covers A labeling of the configuration as to which concepts the near-synonyms actually cover, or lexicalize.

Periph A set of peripheral concepts, which extend the core denotation, and pertain to the differentiation of the near-synonyms.

P-link A set of variables and constraints that 'link' the core denotation to the peripheral concepts.

Distinctions A set of distinctions between the near-synonyms.

For example, listing 7.5 shows the actual representation of the *error* cluster in our system. In the following sections, we will use this example, and others, in describing both the model and implementation of each of the fields.

7.3.1 Syns

Each cluster has a list of its near-synonyms. The near-synonyms are identified by the names of their lexical entries. Lexical entries each contain a syntactic–semantic template that links the conceptual roles in the core denotation of the word to the surface-level semantic roles, and that specifies how the word combines with others. Lexical entries are discussed in section 7.4 below.

7.3.2 Core

In our model, the core denotation serves two main purposes. First, it is the essential language-neutral context-independent denotation of each of the near-synonyms in the cluster. Second, it serves to link the near-synonyms to the ontology, so that it can be used as the necessary applicability condition of all of the near-synonyms in the cluster. (That is, in generation, it must match part of the input to the system in order for any of the words to be available for choice. And in analysis, it is the portion of meaning that is necessarily added to the meaning representation of an utterance.) In this respect, the core denotation has the same function as in the conventional model; the only difference is that it is shared by the near-synonyms of the cluster.

As in MOOSE, and some other knowledge-based systems (Horacek, 1990; Elhadad, McKeown and Robin, 1997; Beale et al., 1998), we represent a denotation, and hence a core denotation, as a configuration of concepts and relations. Configurations allow for a flexible linking of words to concepts, so we can mediate between levels of granularity in the ontology and the lexicon. Thus, we can both maintain a language-neutral ontology and not restrict the model to one-to-one word–concept links.

A configuration is a directed graph of concepts linked by relations. Configurations can be of arbitrary size, from a single concept up to any number of interrelated concepts. For instance, the core denotation of the *error* cluster is simply one concept—a traditional direct link to the ontology (see figure 7.4 (i)). But the denotation of the *lie* 'to tell a lie' cluster (in figure 7.4 (ii)) involves three concepts, which can be interpreted as a activity in which a sayer communicates

```
(defcluster error_C
    ;;; from Gove
    :syns (error_l mistake_l blunder_l slip_l lapse_l howler_l)
    :core (ROOT Generic-Error)
    :covers (ROOT)
    :p-link ((V1 (:and (Person V1) (ACTOR ROOT V1)))
             (V2 (:and (Deviation V2) (ATTRIBUTE ROOT V2))))

    :periph ((P1 Stupidity (ATTRIBUTE-OF V1))
             (P2 Blameworthiness (ATTRIBUTE-OF V1))
             (P3 Criticism (ACTEE V1) (ATTRIBUTE (P3-1 Severity)))
             (P4 Misconception (CAUSE-OF V2) (ACTOR V1))
             (P5 Accident (CAUSE-OF V2) (ACTOR V1))
             (P6 Inattention (CAUSE-OF V2) (ACTOR V1)))

    :distinctions (
     ;;  Blunder commonly implies stupidity.
     (blunder_l usually medium implication P1)

     ;;  Mistake does not always imply blameworthiness, blunder sometimes.
     (mistake_l sometimes medium implication (P2 (DEGREE ’medium)))
     (blunder_l sometimes medium implication (P2 (DEGREE ’high)))

     ;;  Mistake implies less severe criticism than error.
     ;;  Blunder is harsher than mistake or error.
     (mistake_l always medium implication (P3-1 (DEGREE ’low)))
     (error_l always medium implication (P3-1 (DEGREE ’medium)))
     (blunder_l always medium implication (P3-1 (DEGREE ’high)))

     ;;  Mistake implies misconception.
     (mistake_l always medium implication P4)

     ;;  Slip carries a stronger implication of accident than mistake.
     ;;  Lapse implies inattention more than accident.
     (slip_l always medium implication P5)
     (mistake_l always weak implication P5)
     (lapse_l always weak implication P5)
     (lapse_l always medium implication P6)

     ;;  Blunder expresses a pejorative attitude towards the person.
     (blunder_l always medium pejorative V1)

     ;;  Blunder is a concrete word, error and mistake are abstract.
     (blunder_l high concreteness)
     (error_l low concreteness)
     (mistake_l low concreteness)

     ;;  Howler is an informal term
     (howler_l low formality))
    )
```

Listing 7.5: The representation of the *error* cluster.

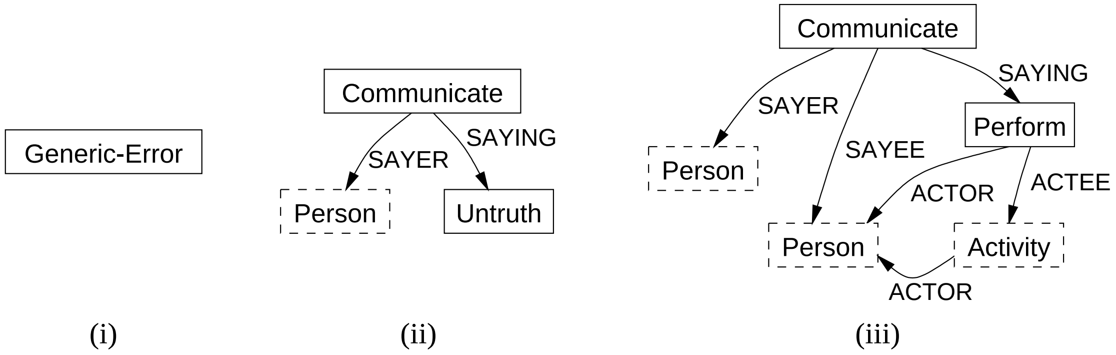


Figure 7.4: Core denotations of the *error* (i), *lie* (ii), and *order* (iii) clusters.

core	::=	(*root-var type relation∗*)
relation	::=	(*role configuration*)
configuration	::=	*variable* |
		(*variable type*) |
		(*variable relation+*) |
		(*variable type relation+*) |
		relation+
type	::=	⟨any concept defined in the ontology⟩
role	::=	⟨any relation defined in the ontology⟩
variable	::=	⟨any symbol⟩
root-var	::=	`ROOT`

Figure 7.5: Syntax of a core denotation.

an untruth. (The dashed box indicates that the concept `Person` is not covered by the near-synonyms in the cluster; see the next section on covering.) Finally, the denotation of the *order* cluster (in figure 7.4 (iii)) states (in a simplistic manner) that it is a communication from one person to a second person of an activity that the second person must perform.

The model allows for even larger denotational structures, which are required to model events and other complex phenomena (see Stede (1998) or Elhadad, McKeown and Robin (1997)). Alternatively, we could have defined each of the core denotations above by single concepts rather than by complex configurations. That is, we could have defined in the ontology a concept to represent 'making an error' or 'communicating an order to another', but that is a decision that should be made by a lexicographer or knowledge engineer, who would decide whether or not the concept is language-neutral.[2] In any case, the model allows for both options.

[2] Note also that description logics (and Loom) impose some practical restrictions. Most description logics cannot represent graph-structured concept definitions, only tree-like definitions. Thus, the denotation of *order* could not be represented in the ontology even if one decided it was language-neutral.

Implementation Figure 7.5 shows the syntax for specifying core denotations. The denotations of the *error, lie,* and *order* clusters, in this language, are as follows:

error:

```
(ROOT Generic-Error)
```

err:

```
(ROOT Communicate
      (SAYER (V1 Person))
      (SAYING (V2 Untruth)))
```

order:

```
(ROOT Communicate
      (SAYER (V1 Person))
      (SAYEE (V2 Person))
      (SAYING (V3 Perform (ACTOR V2)
                          (ACTEE (V4 Activity (ACTOR V2))))))
```

This syntax provides for the minimal representation of denotations,[3] but it is more expressive than Loom, because it allows graph-structured concepts to be defined. This is made possible by identifying each node with a unique variable. For instance, in the denotation of the *order* cluster, the sayee is coreferential with the actor of the ordered activity, as indicated by the variable V2.

The ROOT variable has a special status. It 'coerces' the graph into a tree-like structure, with the ROOT concept as root. If we did not give one node this special status, then we could not as easily connect the denotation to the ontology (i.e., computing subsumption would be intractable), and matching conceptual configurations to one another and to meaning representations would become a potentially complex, even intractable, computational problem. The same decision has been made in many knowledge-based systems from MUMBLE (McDonald, 1983) up to current systems such as MOOSE (Stede, 1999) and Hunter-Gatherer (Beale et al., 1998).

Variables serve several other important functions. They link the concepts in the core denotation to the peripheral concepts, which extend the core denotation. They also link the uncovered concepts to their surface-level roles, defined in the individual lexical entries. For instance, the variable V1 in the *order* denotation links the sayer of the order to the subject role of the verb *order* (see section 7.4 below).

7.3.3 Covers

When we represent word denotations as configurations of concepts, we can introduce the idea of *covering*. That is, if the core denotation of a word exactly matches the structure and content (modulo the subsumption of concepts and relations) of a portion of the input representation, which is represented as a configuration of instances of concepts, then the word is said to *cover* the nodes in that portion of the input. (Thus, a word is applicable only if it completely covers part of the input.) The goal of many sentence generation systems, including those developed by Nogier and Zock (1992) and Nicolov, Mellish and Ritchie (1996), was to find the set of words

[3] In MOOSE, Stede includes atomic values (i.e., values not defined as concepts), negations of atomic values, and default values. Essentially, any kind of additional formalism that one deems necessary to encode denotations could be added to the language.

that collectively maximized coverage of the input. However, in these systems it was always assumed that every node of the denotation participated in covering the input. Stede (1999), and earlier, DiMarco, Hirst and Stede (1993), showed that this is not always the case: a word will not necessarily lexicalize, or cover, all of the concepts in its denotation. For instance, in the denotation of the *lie* cluster above, only the concept of 'communicating an untruth' and not the person who actually tells the lie is lexicalized by the verb *err* (or its near-synonyms). That is, *lie* covers the ROOT and V2 nodes; the V1 node must be lexicalized separately, but it is nevertheless part of the denotation of *lie*.

Thus, in order for a generation system to be capable of completely lexicalizing its input, every word denotation must include information on which nodes of the input it will cover. In this scheme, a word is applicable only if its *whole* core denotation exactly matches part of the input, irrespective of whether it completely covers the same part of the input. Generally, nodes that refer to the 'external' participants of a situation denoted by a word are not covered, whereas nodes that refer to the internal aspects (i.e., what is usually considered part of a denotation in other formalisms) are covered. A covering is a labeling of nodes in the graph. In figure 7.4, solid boxes indicate covered nodes, and dashed boxes, uncovered nodes.

Implementation Following MOOSE, the 'covers' field of a cluster contains the list of variables that refer to the concepts in the core denotation that are covered.

7.3.4 Periph

As we said in chapter 5, the peripheral concepts are concepts that pertain to the conceptual differentiation of the near-synonyms in a cluster. In one sense, defining a set of peripheral concepts serves a practical purpose in making the representation of denotational distinctions more efficient, because they save the re-representing of the same information in different distinctions. But peripheral concepts also have a more fundamental purpose in the model.

A peripheral concept is represented by a configuration of concepts that *extends* the core denotation. Thus, it must have a relation to at least one concept in the core denotation. Figure 7.6 shows the core denotation of the *error* cluster extended with three (of the six) peripheral concepts of the *error* cluster: 'stupidity', 'criticism', and 'misconception'. Figure 7.2, page 97, shows the whole cluster, and figure 7.1 includes similar extensions of the core denotation of the *order* cluster.

Because we represent peripheral concepts as extensions of the core concept, we can treat them differently from the core concept (i.e., not as truth-conditions), but we can still use the inference and reasoning mechanisms of the knowledge representation system. This makes several tasks possible, such as matching portions of an input meaning representation (in generation) to lexical distinctions in order to determine which near-synonyms might convey the input; determining whether the input meaning representation is consistent (i.e., does not ask for conflicting nuances); and determining how the peripheral concepts are related to or depend on one another.

A peripheral concept is also a dimension, whether binary, continuous, or multi-valued. So, a near-synonym can have only one 'value' per peripheral concept, which is specified in a distinction involving the near-synonym. This reflects the dimensional nature of most denotational types of variation that we observed in chapter 3. It also allows the system, if necessary, to reason about relationships between peripheral concepts, and to rule out some input meaning representations that ask for conflicting or incompatible nuances to be expressed (e.g., of both low

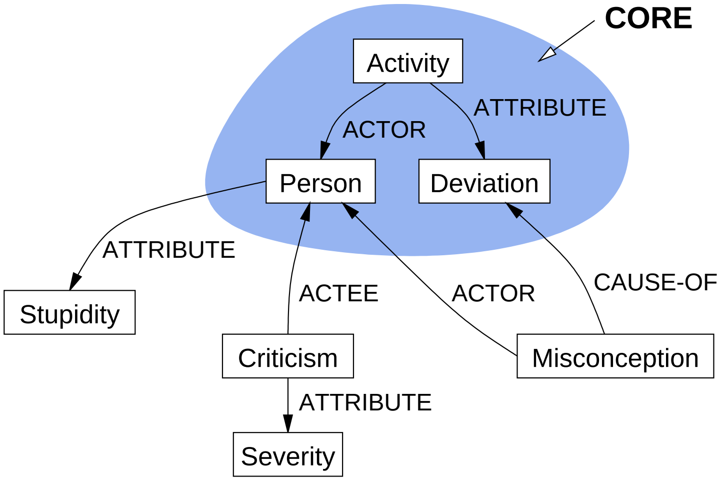


Figure 7.6: Three peripheral concepts related to the core denotation of the *error* cluster.

```
periph          ::=  ( periph-concept∗ )
periph-concept  ::=  ( variable type relation∗ )
relation        ::=  ( role configuration )
configuration   ::=  variable |
                     ( variable type ) |
                     ( variable relation+ ) |
                     ( variable type relation+ ) |
                     relation+
type            ::=  ⟨any concept defined in the ontology⟩
role            ::=  ⟨any relation defined in the ontology⟩
variable        ::=  ⟨any symbol⟩
```

Figure 7.7: Syntax of the ‘periph’ field and of peripheral concepts.

and high severity of criticism, or of both official and peremptory authority). So, it is crucial that the lexicographer specify only peripheral concepts that act as true dimensions.

Implementation Figure 7.7 shows the syntax of the ‘periph’ field. It consists of a list of definitions of peripheral concepts. A peripheral concept is, at minimum, a single ontological concept with one relation to a concept in the core, but it can be an arbitrary-sized configuration of concepts. For example, the six peripheral concepts that we identified for the *error* synonyms are the following:

```
((P1 Stupidity (ATTRIBUTE-OF V1))
 (P2 Blameworthiness (ATTRIBUTE-OF V1))
 (P3 Criticism (ACTEE V1) (ATTRIBUTE (P3-1 Severity)))
 (P4 Misconception (CAUSE-OF V2) (ACTOR V1))
```

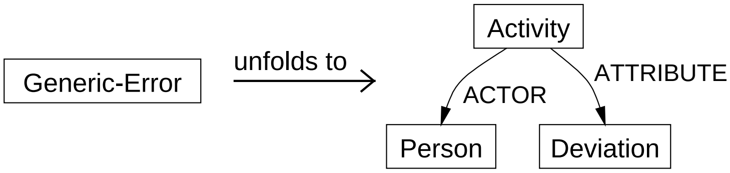


Figure 7.8: The core denotation of the *error* cluster unfolded using the ontology.

p-link	::=	(*var-con*∗)
var-con	::=	(*variable constraint*)
variable	::=	⟨any symbol⟩
constraint	::=	⟨a Loom yes / no query⟩

Figure 7.9: Syntax of the 'p-link' field.

```
     (P5 Accident (CAUSE-OF V2) (ACTOR V1))
     (P6 Inattention (CAUSE-OF V2) (ACTOR V1)))
```

Each has a unique variable (e.g., P1) that will be used in the distinctions to refer to the concept. Also, notice that the configurations both relate the peripheral concept to the core denotation (e.g., P1 is an attribute of the actor of the error), and define the concept (e.g., P3 is a concept of severity of criticism.

7.3.5 P-link

In the cluster, we often need to refer to some concepts that are not explicitly specified in the core denotation, but are related to concepts in the denotation because of the ontology. For instance, in the *error* cluster, we need to refer to the person who commits the error, and to the deviation involved in the error, but neither of these concepts is represented in the core denotation of the cluster. They are, however, referred to in the definition of the Generic-Error concept in the ontology. We can use the ontology to 'unfold', or expand, the core denotation, as shown in figure 7.8, so that we can refer to other concepts in the cluster.

Implementation The 'p-link' field, which 'links' the core denotation to the peripheral concepts through variables, declares a set of new variables that refer to concepts not specified in the core denotation. Figure 7.9 shows the syntax for the field. Each declaration consists of a variable name and a constraint on its value, which generally includes its type (i.e., concept) and relations to other concepts in the core denotation. For instance, the following declaration, from the *error* cluster, states that V1 refers to the concept of the actor (who is a person) of the error, and V2 refers to the deviation attribute of the error.

```
    ((V1 (:and (Person V1) (ACTOR ROOT V1)))
     (V2 (:and (Deviation V2) (ATTRIBUTE ROOT V2))))
```

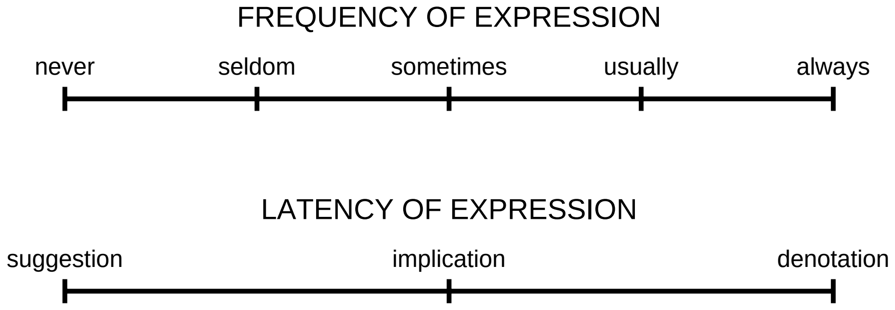


Figure 7.10: The continuums of frequency and indirectness of expression.

Constraints are specified as standard Loom yes / no queries, with variables in the place of the instances (of concepts) that one normally puts in queries. During processing, the queries can be evaluated either to give values to the variables or to check the well-formedness of any instance of a concept bound to a variable.

7.3.6 Distinctions

As we argued in section 5.3, although we would like to represent differences explicitly as first-class objects (so that we can reason about them during processing), we can't do so if we are to have a practical and efficient means of representing the differences. The solution was to represent differences implicitly, but to provide a method for computing explicit differences as needed. Thus, we associate with each near-synonym in a cluster a set of distinctions that are taken to be relative within the cluster; the cluster establishes their local frame of reference. So, a word's set of distinctions implicitly differentiates the word relative to its near-synonyms. (This approach, in effect, situates the word in a multi-dimensional space relative to its near-synonyms, a point of view that we capitalize on in our lexical choice algorithm in section 9.5.)

Now, following our analysis of synonymic variation (chapter 3), we model, in the following sections, three kinds of distinction—denotational, expressive, and stylistic.

Representing denotational distinctions

In chapter 3 we determined three major ways in which words can differ in expressing denotational meaning. Near-synonyms can differ in the frequency with which they express an idea (as influenced by the context), in the indirectness of the expression of the idea, and in fine-grained variations of the idea itself. Thus, to mirror the usage notes, we model a denotational distinction as a list of five components:

(LEXICAL-ITEM FREQUENCY-OF-EXPRESSION STRENGTH INDIRECTNESS CONCEPT)

The first component is the lexical item to which the distinction pertains. The second is a value on the continuum of frequency of expression. The combination of the third and fourth components form a value on the continuum of indirectness of expression. Rather than use numeric values in our formalism—in order to keep it distinct from and more general than any particular implementation—we use a set of symbols that represent positions on the continuums. Figure 7.10 shows the two continuums with their range of values. Frequency can take any of the five values shown. Indirectness (the fourth component) can take three different values, but this is too crude to represent some distinctions (for instance, two words can differ in their *strength* of suggestion), so a value of strength (either weak, medium, or strong) can be specified to shift up or down the value of the indirectness, as long as the indirectness isn't 'denotation', which cannot vary in strength. (This assumes that strength is related to indirectness; for instance, that the stronger a suggestion is, the less indirect it is.) The fifth component is a concept that distinguishes the word from its near-synonyms. It is specified as a configuration of concepts, like all other concepts in the cluster, but it must be a refinement of one of the peripheral concepts. That is, it must extend one of the peripheral concepts by adding new relations and concepts to one or more of the (ontological) concepts in the peripheral concept's definition. This follows from our requirement that peripheral concepts be treated as dimensions.

The configuration can refine a peripheral concept in five different ways, depending on the nature of the peripheral concept. Examples of each type are given in listing 7.6 below, after we give the formal syntax of denotational distinctions, also below.

- As a binary dimension, the configuration can simply specify that a word might express the peripheral concept. Combined with the frequency of the distinction, it can specify a range from never expresses to always expresses the concept. (In this case, the peripheral concept is not really refined at all.)
- As a continuous dimension, it can be refined by a numeric value that represents a position on the dimension. (Since numeric values might be too precise and ad hoc, the next option would usually be used instead.)
- Also as a continuous dimension, it can be refined by a fuzzy set of values, which has the effect of specifying a *fuzzy concept* (see below).
- As a multi-valued dimension, it can be refined by a specific concept that is related to the peripheral concept in some way (e.g., through an attribute relation).
- All of these refinements can be combined into an arbitrary-sized extension of a peripheral concept.

Of course, a peripheral concept can only be extended according to the constraints imposed by the ontology. So, for example, only a `Scalable` concept (i.e., one that can have a `DEGREE` relation) can be extended with a numeric or fuzzy value.

Implementation As shown in figure 7.11, denotational distinctions are represented by 5-tuples that include the lexical item, frequency, strength, indirectness, and conceptual configuration of the distinction. Because the conceptual configurations are extensions of peripheral concepts, there is no need to retype the whole peripheral concept. Instead, one need only specify the additional parts of the configuration with, of course, a relation back to the peripheral concept (by using a variable declared in the peripheral concept).

Listing 7.6 shows a few examples of how various types of denotational distinction are represented in the system. The `Pn` variables refer to nodes in the peripheral concepts. The `X` variables represent dummy variables on configurations that aren't referred to anywhere else in the cluster; they are not co-referential. For reference, the complete representations of the clusters are given in appendix B.

```
distinctions              ::=  ( distinction∗ )
distinction               ::=  denotational-distinction | expressive-distinction | stylistic-distinction

denotational-distinction  ::=  ( lexitem frequency strength indirectness configuration )
lexitem                   ::=  ⟨any member of the ‘syns’ field⟩
frequency                 ::=  never | seldom | sometimes | usually | always
strength                  ::=  weak | medium | strong
indirectness              ::=  suggestion | implication | denotation

expressive-distinction    ::=  ( lexitem frequency strength attitude variable )
attitude                  ::=  pejorative | neutral | favourable

stylistic-distinction     ::=  ( lexitem degree style )
degree                    ::=  low | medium | high
style                     ::=  formality | force | concreteness | floridity | familiarity | ···

configuration             ::=  value |
                               variable |
                               ( variable type ) |
                               ( variable relation+ ) |
                               ( variable type relation+ ) |
                               relation+
relation                  ::=  ( role configuration )
type                      ::=  ⟨any concept defined in the ontology⟩
role                      ::=  ⟨any relation defined in the ontology⟩
variable                  ::=  ⟨any symbol⟩
value                     ::=  numeric-value | fuzzy-value
numeric-value             ::=  ⟨0..1⟩
fuzzy-value               ::=  ⟨a fuzzy set on 0..1⟩
```

Figure 7.11: Syntax of the ‘distinctions’ field. (The syntax of each of the three major kinds of distinctions is shown.)

Distinctions on a binary dimension

```
;; Blunder usually implies stupidity.
(blunder_l usually medium implication P1)

;; Escort usually suggests the action is a courtesy.
;; Attend sometimes suggests the action is a courtesy.
;; Convoy never suggests the action is a courtesy.
;; NOTE: P3 -> (P3 Courtesy (ATTRIBUTE-OF ROOT))
(escort_l usually medium suggestion P3)
(attend_l sometimes medium suggestion P3)
(convoy_l never medium suggestion P3)
```

Distinctions on a continuous dimension

```
;; Mistake always implies a low degree of severity of criticism.
;; Error always implies a medium degree of severity of criticism.
;; Blunder always implies a high degree of severity of criticism.
(mistake_l always medium implication (P3-1 (DEGREE ’low)))
(error_l always medium implication (P3-1 (DEGREE ’medium)))
(blunder_l always medium implication (P3-1 (DEGREE ’high)))

;; Forest always denotes a high (degree of) size.
;; Woods always denotes a low to medium size.
;; NOTE: P1 -> (P1 Size (ATTRIBUTE-OF ROOT))
(forest_l always medium denotation (P1 (DEGREE ’high)))
(woods_l always medium denotation (P1 (DEGREE ’(OR low medium)))
```

Distinctions on a multi-valued dimension

```
;; Command always implies official authority.
;; Order always implies peremptory (arbitrary) authority.
;; NOTE: P1 -> (P1 Authority (ATTRIBUTE-OF V1))
(command_l always medium implication (P1 (ATTRIBUTE (X Official))))
(order_l always medium implication (P1 (ATTRIBUTE (X Peremptory))))

;; Accompany usually implies equal status.
;; Attend usually implies a subordinate status of the attender.
;; NOTE: P2 -> (P2 Status (ATTRIBUTE-OF V1))
(accompany_l usually medium implication (P2 (ATTRIBUTE (X Equal))))
(attend_l usually medium implication (P2 (ATTRIBUTE (X Subordinate))))
```

Listing 7.6: Examples of various denotational distinctions represented in the system.

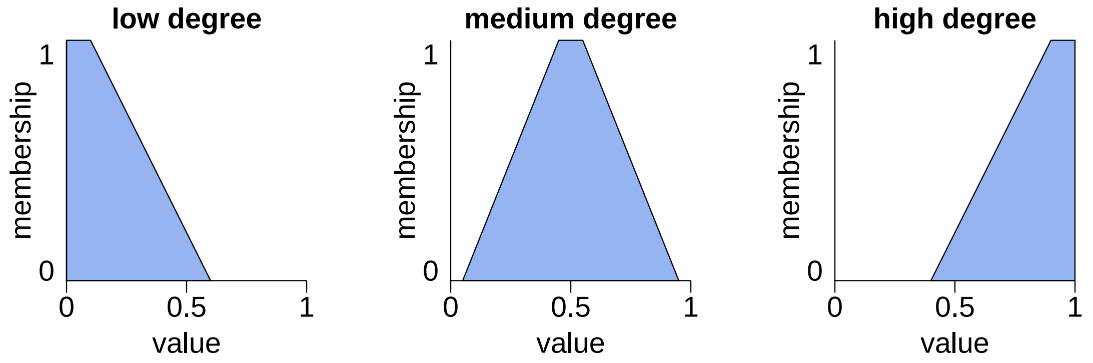


Figure 7.12: Membership functions for the fuzzy sets of low, medium, and high degree.

Fuzzy distinctions A *fuzzy set* is a set in which each member has an associated degree of membership, as defined by a membership function. In our model, we use fuzzy sets to represent fuzzy degrees of severity, size, or any other scalable concept. For instance, `low` represents a fuzzy area at the low end of a dimension so that a range of values are all considered low, rather than just a single value such as zero.

We implemented fuzzy sets in the standard way described by Dubois and Prade (1980). The membership function takes a member of the domain, in our case a position on a dimension, and returns a value in $[0,1]$. Figure 7.12 shows the membership functions for three fuzzy sets that represent `low`, `medium`, and `high` degrees, respectively. The domain of the fuzzy sets is the set of possible values of the `DEGREE` relation, that is, $[0,1]$. For instance, a value of 0.5 in the domain has a degree of membership of 0.2 in `low`, 1.0 in `medium`, and 0.2 in `high`.

We also implemented two operators on fuzzy sets: union and conformity. The union of two fuzzy sets is the fuzzy set whose membership function is the maximum (over the domain) of the membership functions of the two sets (i.e., the union of the areas under their membership functions). For instance, `(OR low medium)` is the union of `low` and `medium` and represents a wider range of value at the low end of the dimension.[4]

The conformity of two fuzzy sets is a value that indicates how well two fuzzy sets match or overlap one another. There are many ways to compute conformity (see Dubois and Prade (1980)), but we implemented the conventional method. The conformity of two fuzzy sets is the height of their intersection, which is, in turn, the area of overlap under their membership functions. For instance, the conformity of `low` and `medium` is 0.6, and of `low` and `high` is 0.2. We can also determine the conformity of a numeric value and a fuzzy set.

By using fuzzy sets in our representations we can, in effect, model fuzzy distinctions and use fuzzy matching to compare meaning representations to lexical representations. This idea is similar to the use of fuzzy sets in fuzzy database systems (Li and Liu, 1990). In a fuzzy database, a database record, which consists of a set of attribute / value pairs, can have a fuzzy set as a value for one or more of its attributes. For instance one could represent someone's age as YOUNG, OLD, and VERY-OLD, which are fuzzy sets on the domain of age (i.e., $[0,120]$). Then, a database query that asks for records of people who are 80 years old would return various records with matching scores that depend on how well the value 80 conforms to the fuzzy set values of each

[4] We could define other ways to create new fuzzy sets, such as `(VERY low)`, `(SORT-OF low)`, and `(NOT low)`, but we haven't done so yet.

	Peripheral concept		
Word	Size	Wildness	Urban proximity
forest	`high`	`high`	`low`
woods	`(OR low medium)`	`low`	`(OR medium high)`
Wald (Ger.)	`(OR medium high)`	`(OR medium high)`	`(OR low medium)`
Gehölz (Ger.)	`low`	`low`	`high`

Table 7.1: Values on the dimensions of the *forest* cluster for some of its near-synonyms.

database record (e.g., 80 would probably conform very well to VERY-OLD, well to OLD, and not at all to YOUNG).

In our model, a lexical entry is like a database record. The peripheral concepts are attributes, and the distinctions provide the values. For example, we could represent the meanings of some of the *forest* words as shown in table 7.1 (see figure 3.5, page 45). Thus, the concepts denoted by *forest* and *woods* are fuzzy and overlap to a certain extent. They would both match to different degrees the same meaning representation of a particular tract of trees. And clearly, the German *Wald* could translate reasonably well to either *woods* or *forest*, because of fuzzy matching, if no other information about the referent of *Wald* were known.

Representing expressive distinctions

Our model can represent differences in the attitude that the speaker takes towards an entity of the situation as conveyed by a word. We take a conservative approach, pending future research, and define only three possible attitudes: favourable, neutral, and pejorative. However, the attitude can be modified by a strength parameter as in the denotational distinctions above. As well, whether or not a word conveys an attitude depends on the context, so like denotational distinctions, we include a frequency value. Finally, since a word can express an attitude towards potentially any participant of a situation, we must include a reference to the participant.

Thus, an expressive distinction has the following form:

(LEXICAL-ITEM FREQUENCY STRENGTH ATTITUDE ENTITY)

where the first component is the lexical item, the second is the frequency with which the attitude is conveyed by the lexical item, the third and fourth, combined, are the attitude taken by the speaker, and the fifth is a reference to one of the concepts in the core denotation (or ‘p-link’ field).

Implementation Figure 7.11 shows the syntax for representing expressive distinctions. In order to refer to the entity towards whom the attitude is directed, we simply use the variable that refers to the entity’s concept. For example, *blunder* can convey a pejorative attitude towards the person who commits the blunder, whose variable is `V1` in the cluster. So, the representation of this attitude is

```
(blunder_l always medium pejorative V1)
```

In the cluster of *thin* adjectives, we represent the following expressive distinctions, where V1 refers to the person who is thin:

```
(skinny_l usually medium pejorative V1)
(thin_l usually medium neutral V1)
(slim_l usually medium favourable V1)
(slender_l usually strong favourable V1)
```

And, finally, in the following distinctions, the attitudes are made towards the referent of the word itself, referred to by the ROOT variable:

```
(drunk_l always strong pejorative ROOT)
(drunkard_l always medium pejorative ROOT)
(alcoholic_l always medium neutral ROOT)
(tippler_l always medium favourable ROOT)
```

Representing stylistic distinctions

Since we don't yet have a comprehensive theory of style, we take a rather basic approach to representing the stylistic variation of near-synonyms, similar to past approaches. Unlike the denotational distinctions above, stylistic features have a global or absolute quality to them. We can compare all words, whether or not they are near-synonyms, on various stylistic dimensions, such as formality and concreteness. Because style is a global aspect of text, a certain style can be (and should be) achieved by more than just lexical choice; structural choices are just as important. Hence, in defining a set of stylistic dimensions, we must look for global stylistic features that can be carried not only by words, but also by syntactic and larger text structures.

Stylistic variation is different from the other types of variation in another respect. It is related to the form of a word and not to its denotation or conceptual meaning, though in a deeper sense style is certainly related to meaning. So, in representing stylistic distinctions we don't have to make any reference to entities or aspects of the core denotation or peripheral concepts in a cluster.

Our stylistic dimensions include, but are not limited to, the following: *formality*, *force*, *concreteness*, *floridity*, and *familiarity* (see Hovy (1988) and Stede (1993)). And the value a word has on any of these dimensions is taken from the set of *low*, *medium*, and *high* (though more values could easily be added to increase the precision.)

Implementation Figure 7.11 shows the syntax for representing stylistic distinctions. A distinction is simply a 3-tuple of lexical item, value, and stylistic dimension. For example, the *error* synonyms differ with respect to their concreteness and formality:

```
(error_l low concreteness)
(mistake_l low concreteness)
(blunder_l high concreteness)

(howler_l low formality)
(error_l medium formality)
(mistake_l medium formality)
(blunder_l medium formality)
```

Default distinctions

For whatever reason, it might not always be known how a near-synonym differs from the others in a cluster. In such cases, we could assume a default distinction on each of the dimensions, but this might not always be the right thing to do. For instance, if it were not known how the word *lapse* stood in relation to its near-synonyms on the dimension of severity of criticism, should we use a default that says it has medium severity? Or no severity? Or no criticism? Clearly, the answer is not straightforward, at least for denotational distinctions. So, in the system we have opted to assume that if a denotational distinction on a particular peripheral concept has not been specified for a particular word, then the word's 'value' on the dimension is unknown. This means that the system cannot compare the word to any other word on that dimension (so if having a particular value on the dimension were a deciding factor in lexical choice, then the word would never be chosen if there were other words that had values of any sort on the dimension).[5]

For stylistic and expressive distinctions, on the other hand, it seems reasonable, and even necessary, to use a default. For instance, if the level of formality was not specified for a particular word, then it would actually be beneficial to the system to assume that the word had a medium formality. That way, the word would still be available, though less preferred, for choice when, say, highly formal words were desired. Likewise, it would be beneficial to assume that a word expresses a neutral attitude towards a participant of the situation, unless otherwise specified. Therefore, the default stylistic distinction for the lexical item *lex* and dimension *stylistic-dimension* is

(*lex* `medium` *stylistic-dimension*)

And the default expressive distinction about the entity referred to by *V* is

(*lex* `always medium neutral` *V*)

7.4 Lexical entries

Every word sense has a lexical entry, which serves two functions. It links a lexical item to its surface form or spelling, and it represents the meaning of the lexical item on a syntactic–semantic level. That is, it specifies how the word can be combined with other words to form larger structures. A range of structural associations can be represented at this level from standard compositional meaning to lexical collocations and idiomatic combinations.

To do this, each lexical entry specifies a template, or case frame, for the lexical item. The template links entities or participants in the core denotation of the word (i.e., the concepts that are not covered) to their surface level roles. For instance, the template for the *order* verb includes three roles that link to the sayer, sayee, and saying of the activity, and that represent how and where they should appear in the surface form. Since near-synonyms can vary in how they map participants to surface roles, and in their collocations, each near-synonym has its own template. (Note that we have not yet modelled collocational behaviour, but if we were to do so, it would be encoded in the lexical entries.)

During generation, a set of templates for the words that have been chosen are unified into a complete and well-formed syntactic–semantic representation, which is then given as input

[5] This might not always give the desired behaviour. If we ask for, say, high severity when there are two options to choose from, one of which expresses low severity and the other of which is unknown, then the system would choose the former option, because it would find that it is the closest match.

psemspec	::=	(*int-variable* / *um-type mod+*)
mod	::=	*keyword ext-variable* | `:lex` *lex-pointer* | `:name` ⟨string⟩
keyword	::=	⟨any Upper Model keyword⟩
int-variable	::=	`x` | `x1` | `x2` ···
ext-variable	::=	⟨any variable in the core denotation⟩
um-type	::=	⟨any concept defined in the Upper Model⟩
lex-pointer	::=	⟨any morphosyntactic entry in the Penman lexicon⟩

Figure 7.13: Syntax of a partial semantic specification (PSemSpec).

to a surface level generator. In analysis, a parser would output a complete syntactic–semantic representation, which could then be mapped to a conceptual representation by using the relationships encoded in the templates.

Implementation Most current generation systems use a representational formalism that can encode the combinatory potential of words at the syntactic–semantic level. Since our focus is not on this level of representation, we provide no new results in this area, and thus adopt MOOSE's formalism without modification. In MOOSE, an expression at this level of representation is called a semantic specification, or SemSpec, and the language for representing it is a subset of SPL, so Penman can generate a sentence from a SemSpec directly. Templates associated with lexical items are thus partial semantic specifications, or PSemSpecs. Stede's syntax for PSemSpecs is repeated in figure 7.13. A PSemSpec must include a link to a lexical entry, in Penman's lexicon, which contains the spelling and morphosyntactic properties of the word. It must also make reference to each uncovered concept in the core denotation by specifying an appropriate Upper Model keyword (such as `:actor` or `:range`) and the (external) variable associated with the uncovered concept; this links the conceptual roles to the surface roles. Internal variables are used for coreference within the completed SemSpec.

For example, listing 7.7 shows the lexical entries for a few of the words referred to in this chapter. Each entry includes the language of the word, the cluster to which it belongs, and its PSemSpec.

7.5 Summary

In this chapter, we presented a clustered model of lexical knowledge that is intended to account for many of the types of lexical variation that we classified in chapter 3. The model is based on the conventional ontological model, but it cuts off the ontology at a coarse grain and differentiates near-synonyms on a subconceptual / stylistic level of semantics. Every cluster of near-synonyms has a core denotation, which represents the essential language-neutral denotation shared by its near-synonyms, and which acts as a necessary applicability condition of each of the near-synonyms. The near-synonyms are implicitly differentiated; each has a set of lexical distinctions that, in effect, situate the word in a multi-dimensional space relative to its near-synonyms. Because of the properties of the model, which we listed at the beginning of this chapter, we feel that the model overcomes the problems of other current models in regards to representing near-synonymy (see section 4.3), and so is able to adequately account for near-synonymy.

```
(deflex error_l
    :cluster error_C
    :language english
    :psemspec (x / Abstraction :lex error))

(deflex forest_l
    :cluster forest_C
    :language english
    :psemspec (x / Three-D-Location :lex forest))

(deflex err_l
    :cluster err_C
    :language english
    :psemspec (x / Nondirected-Action :lex err :actor V1))

(deflex lie-v_l
    :cluster tell-a-lie_C
    :language english
    :psemspec (x / Nondirected-Action :lex lie-v :actor V1))

(deflex order_l
    :cluster order_C
    :language english
    :psemspec (x / addressee-oriented-verbal-process :lex order
                 :sayer V1 :addressee V2 :saying V4))

(deflex thin_l
    :cluster thin_C
    :language english
    :psemspec (x / Sense-And-Measure-Quality :lex thin))
```

Listing 7.7: Examples of lexical entries represented in the system.

Table 7.2 is a reproduction of table 3.14 that indicates the types of variation that our model is intended to account for. Solid circles (●) indicate that the model accounts for the variation, and that we implemented it. Semi-solid circles (◉) indicate that, while the model should be able to account for the type of variation, we have yet to incorporate it into the model. And open circles (○) indicate that more research is required to determine both whether the model can accommodate the type of variation and how to do so, though we are confident that the model is general enough to allow this.

Specifically, we can represent stylistic variation between words, but not variation in dialect or sublanguage. The latter would seem to pose no significant problem beyond the problems associated with stylistic dimensions. We would need to define a set of dialects and sublanguages, but it is more likely that separate lexicons would be developed for different dialects, since it is not immediately clear what sort of application would need to use a word in multiple dialects at once.

While we can represent variation in expressed attitude, we did not account for other emotive effects of words. Very little research has been done in this area, almost none in computational linguistics, but work by Mel'čuk and Wanner (1994) on emotive verbs is a beginning.

We can also represent variation both in the manner of expression of denotational meaning and in the meaning itself. We have not yet considered emphasis, since we lack a formal theory of the phenomenon. Clearly, emphasis is a relativistic and context-dependent phenomenon, so to account for it, we will have to confront the issue of how context influences lexical meaning, one of the current major open problems in computational linguistics. Another major open problem, that of using world knowledge for effective inference, also caused us to leave a hole in the model in not accounting for complex 'dimensions' of variation.

Finally, though we did not concern ourselves directly with collocational issues, the model can accommodate collocational behaviour. Selectional restrictions could be incorporated into the core denotations as conceptual constraints on the fillers of roles (e.g., that a `Person` must fill a particular `ACTOR` role). Collocational constraints (or preferences) could be represented on the syntactic–semantic level in the lexical entries. These could be represented as lexical constraints (or preferences) on the fillers of surface-level roles (e.g., that a particular adjective should be chosen to modify a particular word in a given lexical entry, say *daunting* for *task*, and vice versa).

Variation category	Status	Variation type
Stylistic	◉	Geographical dialect
	◉	Temporal dialect
	◉	Social dialect
	◉	Language
	◉	Sublanguage
	●	Formality
	●	Force
	●	Concreteness
	●	Floridity
	○	Euphemism
	●	Familiarity
	◉	Simplicity
Expressive	○	Emotive
	●	Expressed attitude
Denotational	●	Denotation
	●	Implication
	●	Suggestion
	○	Emphasis (rel. to concept)
	○	Emphasis (rel. to word)
	●	Frequency of expression
	◉	Fine-grained technical
	◉	Abstract dimension
	●	Continuous dimension
	●	Binary dimension
	●	Multi-value dimension
	○	Complex 'dimension'
	○	Specificity
	○	Extensional overlap
	●	Fuzzy overlap
Collocational	◉	Selectional restrictions
	◉	Lexical collocation
	◉	Idiom
	◉	Grammatical collocation
	◉	Subcategorization
	◉	Converse

Table 7.2: The types of lexical variation accounted for by the clustered model of lexical knowledge.

Chapter 8

Lexical similarity

For tasks such as lexical choice, we need to be able to compare the similarities of pairs of near-synonyms. For example, in a transfer-based MT system, in order to translate the French word *bévue* into English, we would compare the similarities of at least the three pairs *bévue* : *error*, *bévue* : *mistake*, and *bévue* : *blunder* and choose the target word whose similarity to *bévue* is highest, subject to any constraints arising from the context. We also need to be able to compare the similarities of each of several near-synonyms to a particular meaning representation. For example, in text generation or interlingual MT, we would want to choose the near-synonym that is most similar to the input meaning representation.[1]

Now, the general problem of measuring the *semantic distance* between between words or concepts is well known. In this century, Wittgenstein (1953) introduced the notion of *family resemblance*—that several things can be related because they overlap with respect to a set of properties, no property being common to all of the words—which Rosch (1978) then used as the basis for the prototype-theory of meaning (see section 4.1.4). Recent research in computational linguistics has focused more on developing methods to compute the degree of semantic similarity between any two words, or rather, between the simple or primitive concepts[2] denoted by any two words. Budanitsky (forthcoming 1999) compares many of the different similarity measures, which variously use taxonomic lexical hierarchies, lexical-semantic networks (such as WordNet or *Roget's Thesaurus*), word definitions in machine-readable dictionaries, large text corpora, or a combination thereof (Morris and Hirst, 1991; Dagan, Marcus and Markovitz, 1993; Kozima and Furugori, 1993; Resnik, 1995; Jiang and Conrath, 1997; Leacock and Chodorow, 1998; Karov and Edelman, 1998). Other approaches seek to find semantically-related words in corpora, and so induce a thesaurus through hierarchical clustering techniques (Pereira, Tishby and Lee, 1993; Church et al., 1994; Grefenstette, 1994; McMahon and Smith, 1996; Schütze, 1998; Lin, 1998).

Unfortunately, none of these methods is helpful in computing the similarity of near-synonyms because the similarity measures lack the required precision. That is, the grain size of each of the measures is such that near-synonyms, being so similar to begin with, are actually all considered to be equally similar. For instance, while most of the methods can determine that, say, *banana* and *apple* are more similar than, say, *banana* and *dog*, they cannot determine that *mis-*

[1] Notice our shift to a terminology of 'similarity' rather than of 'difference' and 'distinction', which we used in the last chapter. A difference is generally thought of as an *object*, and indeed they are objects in the model, even if only implicitly. On the other hand, a similarity is usually thought of as a *value* or a degree (or maybe a state), and that is the idea we'll focus on in this chapter.

[2] By primitive concepts, we mean *named* concepts, or concepts that can be lexicalized by a single word, even though they may be defined in terms of other concepts in an ontology.

take and *error* are more similar than *mistake* and *blunder*. The reasons for this lack of precision are clear. First, taxonomic hierarchies and semantic networks (such as WordNet) inherently treat near-synonyms as absolute synonyms in grouping near-synonyms into single nodes. (In any case, as we argued in sections 4.1.1 and 5.1, it would be difficult if not impossible to taxonomize near-synonyms.) Second, standard dictionary definitions are not usually fine-grained enough (i.e., they define the core meaning but not the nuances of a word), and can even be circular, defining each of several near-synonyms in terms of the other near-synonyms. And third, corpus-based techniques, as we saw in chapter 6, are not yet capable of uncovering the more subtle differences in the use of near-synonyms.

Therefore, we need to develop a more precise measure of semantic similarity. Now, given the clustered model of lexical knowledge that we developed in the chapter 7, in which differences between near-synonyms can be implicitly represented, it seems natural to use these differences to compute either explicit differences or degrees of similarity between near-synonyms. After analyzing the problem, however, we concluded that developing a fully effective similarity measure for near-synonyms would require a substantial amount of basic research, so we've left the task for future work. In this chapter, we will characterize the problem of computing the semantic similarity of near-synonyms and give a solution to only one part of the problem: computing the similarity between individual lexical distinctions. In chapter 9, we will develop a method of computing the similarity between an input meaning representation (i.e., a set of preferences) and a set of lexical distinctions, which will rely on the results presented here.

8.1 Computing the similarity of near-synonyms

In the clustered model of lexical knowledge, a difference between two near-synonyms is encoded implicitly in two sets of relative distinctions (see section 7.3.6). From two such sets of distinctions, one can compute, or build, an explicit difference between two near-synonyms. For instance, consider the representations of the differences between *bévue* and each of *mistake, error,* and *blunder* shown in figure 8.1. (Note that for the purpose of this example we assumed that *bévue* has the same representation as *blunder*; refer back to figure 7.5 (page 105) for the `error_C` cluster.) To build these structures, we first paired matching distinctions (e.g., two distinctions matched if they positioned the two words, relative to each other, on the same dimension; if one word did not have a matching distinction, then the distinction was considered to be unknown). We then merged the members of each pair. If two distinctions had the same value for a particular component, we put that value in the merger; if they had different values, we built a representation of the difference. We adapted Hirst's (1995) notation for conceptual differences: a difference on a particular component is represented in square brackets with the opposing values separated by a slash.

Whatever formalism one uses to represent difference-structures, the problem is not in developing the formalism but in using and comparing two or more actual difference-structures. Equivalently, the problem is in computing a similarity value from a single difference-structure (which can then be easily compared to a similarity value from another difference-structure). Part of the problem comes from the complex structure of distinctions. For instance, is a word that suggests criticism more similar to a word that *strongly* suggests a *medium* degree of criticism or to a word that *weakly* suggests a *high* degree of criticism? It is unclear what sort of relationship holds between the indirectness with which an idea is expressed and the idea itself. A similar problem arises with our use of frequency: is a word that implies blameworthiness more similar to a word that *sometimes implies* it or to a word that *always suggests* it (where a suggestion

```
Diff ("bévue" / "mistake") =
  (( [usually / unknown] [medium / unknown] [implication / unknown]
        (Stupidity (ATTRIBUTE-OF V1)) )
   ( [sometimes / usually] medium implication
        (Blameworthiness (ATTRIBUTE-OF V1)) )
   ( always medium implication
        (Criticism (ACTEE V1) (ATTRIBUTE (Severity (DEGREE [more / ])))) )
   ( [unknown / always] [unknown / medium] [unknown / implication]
        (Misconception (CAUSE-OF V2) (ACTOR V1)) )
   ( [unknown / always] [unknown / weak] [unknown / implication]
        (Accident (CAUSE-OF V2) (ACTOR V1)) )
   ( [usually / always] medium [pejorative / neutral] V1 )
   ( [more / ] concreteness ) )

Diff ("bévue" / "error") =
  (( [usually / unknown] [medium / unknown] [implication / unknown]
        (Stupidity (ATTRIBUTE-OF V1)) )
   ( [sometimes / unknown] [medium / unknown] [implication / unknown]
        (Blameworthiness (ATTRIBUTE-OF V1)) )
   ( always medium implication
        (Criticism (ACTEE V1) (ATTRIBUTE (Severity (DEGREE [more / ])))) )
   ( [usually / always] medium [pejorative / neutral] V1 )
   ( [more / ] concreteness ) )

Diff ("bévue" / "blunder") = ()
```

Figure 8.1: Structures that explicitly represent the differences between *bévue* and each of *mistake*, *error*, and *blunder*.

is weaker than an implication)?

Another part of the problem is that near-synonyms will often differ on seemingly incommensurate dimensions. That is, the distinctions of one near-synonym will often not match those of another near-synonym, leaving no basis for comparison. For instance, in figure 8.1, it would be difficult to compute a value for the degree of similarity (or difference) of *bévue* and *error* when they align on only one semantic dimension out of three (notice all the unknowns in `Diff("bévue" / "error")`). Consider also `Diff("bévue" / "mistake")`: can we even compare an error that is caused by a misconception to an error that is stupid? Or to an error that expresses a pejorative attitude?

A related problem comes up when two near-synonyms differ simultaneously on the same set of dimensions. Consider the near-synonyms of *forest*: is it possible to decide whether a large and wild tract of trees is closer to a small wild one or to a medium-sized non-wild one? In other words, how much of a change in the size of a forest will compensate for an opposite change in its wildness?

As we said above, we won't give a full solution to the problem of computing the similarity of near-synonyms. However, we suggest that part of the solution must lie in that the dimensions of a cluster probably aren't completely incommensurate. That is, the dimensions will usually have interrelationships, which can be exploited when comparing the representations of words. For instance, in the cluster of *forest* near-synonyms, the wildness of a tract of trees is probably related to its size and distance from civilization (which one can infer from one's knowledge about forests and wildlife; e.g., most wildlife tries to avoid people). So, there may be a way of comparing a wild tract of trees to a large tract of trees after all. And in the cluster of *error* near-synonyms, the dimensions appear to be related in similar ways because of their semantics and pragmatics (e.g., `Blameworthiness` is related to `Criticism`, and `Stupidity` is related to expressing a pejorative attitude). Surely these interrelationships influence both what can be coherently represented in a cluster, and how similar near-synonyms are. And presumably such relationships can be represented in the ontology.

In our preliminary solution, we make the assumption that the dimensions of a cluster are independent and that each can be reduced to a true numeric dimension.[3] That way, the set of distinctions of a word situates the word in a numeric multi-dimensional space, in which distances on individual and multiple dimensions can be easily computed and compared. In the next two sections we will develop a function for computing the similarity of two individual lexical distinctions, assuming that the distinctions match (i.e., that they specify relative values on the same dimension.)

8.2 Computing the similarity of distinctions

Assuming that we are given two lexical distinctions that are comparable, then we would like to compute a numeric degree of similarity between them. In this section we describe a function, $Sim : D \times D \rightarrow [0,1]$, for computing the similarity of two lexical distinctions taken from the set D of all possible distinctions that can be represented in a particular cluster. A score of zero means that the distinctions are completely different (or can't even be compared because they do not match at all), and a score of one means that they are equivalent though not necessarily

[3] Actually, numeric values would seem to be necessary at some level of representation. As we've seen, nuances of meaning and style are not always clear-cut but can be vague, fuzzy, and continuously variable. So, using a numerical method would seem to be the most intuitive way of computing similarity, which we have to do to compare and choose appropriate lexical items.

Frequency		Indirectness		Attitude		Strength		Style	
never	0.0	suggestion	2	pejorative	-2	weak	-1	low	0.0
seldom	0.25	implication	5	neutral	0	medium	0	medium	0.5
sometimes	0.5	denotation	8	favourable	2	strong	1	high	1.0
usually	0.75								
always	1.0								

Table 8.1: The functions that map symbolic values to numeric values.

identical, since two structurally different concepts might in fact represent the same meaning (see section 8.3).

The similarity scores given by *Sim* are not meant to be taken in any absolute sense; they are to be compared to one another, to decide, for instance, how near-synonyms differ from one another on a particular dimension. Thus, if several scores for several pairs of distinctions (of two near-synonyms) were somehow combined, the resulting overall similarity score would also be relative to the scores of the other pairs of near-synonyms in the cluster.

So, given two distinctions d_1 and d_2, we must first make sure that they match one another, that is, that they can even be compared. This means first that they have to be of the same type. Then, if they are stylistic distinctions, they must also involve the same stylistic dimension; if they are expressive distinctions, they must also involve the same participant of the denotation (i.e., their variables must refer to the same participant); or if they are semantic distinctions, they must also involve the same peripheral concept (i.e., their conceptual configurations must specify ‘values’, possibly different, on the same dimension). Hence,

$$Sim(d_1,d_2) = \begin{cases} 0.0 & \text{if } d_1 \text{ and } d_2 \text{ do not match} \\ Sim_{semantic}(d_1,d_2) & \text{for semantic distinctions} \\ Sim_{expressive}(d_1,d_2) & \text{for expressive distinctions} \\ Sim_{stylistic}(d_1,d_2) & \text{for stylistic distinctions} \end{cases} \tag{8.1}$$

Now, as we saw in section 7.3.6, distinctions are formed out of several components, which are represented as symbolic values on certain dimensions and continuums. In order to compute a numeric score, we have to map each symbolic value to a corresponding numeric value. Table 8.1 shows the mappings that we used for each of the types of dimension and continuum. Values of *Frequency* are evenly spaced from 0.0 to 1.0. The *Indirectness* continuum runs from 1 to 8, because the *Indirectness* values can be shifted up or down by *Strength* factors (that range from -1 to 1), except that the value *denotation* cannot be shifted. Values of *Attitude* range from -3 to 3, and can also be shifted by *Strength* factors, but because the dimension has a negative end, values are shifted away and towards the origin for “strong” and “weak” modifications, respectively. The numeric values themselves and their possible ranges aren’t as important as their relative differences, since everything gets normalized to the interval $[0,1]$.

As equation (8.1) shows, the three types of distinction each have their own similarity function. (Note that we experimented with several ways of computing similarity, but settled for the simple approach discussed here. Until a comprehensive empirical evaluation can be done, there is no reason to try anything more complex.) For stylistic distinctions, the degree of similarity is one minus the absolute value of the difference between the values, as shown in equation (8.2). So, if the pairs of values are “low” and “medium”, the similarity is, according to table 8.1, $1.0 - |0.0 - 0.5| = 0.5$.

$$Sim_{stylistic}(d_1, d_2) = 1.0 - |Style(d_2) - Style(d_1)| \tag{8.2}$$

For expressive distinctions, similarity depends on the frequencies and on the attitudes. The similarity of two frequency values is one minus their absolute difference (see equation (8.4)). For the attitudes, we shift each of the attitudes by their strength, then perform the same calculation, except that we normalize the result (by dividing by the length of the continuum, 6, in this case). Equations (8.5) and (8.6) show how. Finally we multiply the two results, as equation (8.3) shows. Figure 8.2 gives an example.

$$Sim_{expressive}(d_1, d_2) = S_{freq}(d_1, d_2) \times S_{att}(d_1, d_2) \tag{8.3}$$

$$S_{freq}(d_1, d_2) = 1.0 - |Frequency(d_2) - Frequency(d_1)| \tag{8.4}$$

$$S_{att}(d_1, d_2) = 1.0 - \frac{|Att(d_2) - Att(d_1)|}{6} \tag{8.5}$$

$$Att(d) = Attitude(d) + \operatorname{sgn}(Attitude(d)) \times Strength(d) \tag{8.6}$$

The similarity of two semantic distinctions is the product of the similarities of their three components: frequencies, latencies, and conceptual configurations (see equation (8.7)). (Figure 8.2 gives two examples.) The scores for the first two are calculated in a similar manner to the expressive distinctions. But with the conceptual configurations, we encountered an intriguing (unsolved) problem. Informally, the two distinctions represent a pair of values on the same dimension, so their similarity should be related to how close the values are to each other. But the values, and the dimensions themselves, are represented as conceptual configurations, so computing their closeness is not as simple as applying a distance metric. In fact, we must calculate how similar the whole conceptual configurations are to one another. We discuss the problem of computing *conceptual similarity* in the next section.

$$Sim_{semantic}(d_1, d_2) = S_{freq}(d_1, d_2) \times S_{ind}(d_1, d_2) \times S_{con}(d_1, d_2) \tag{8.7}$$

$$S_{ind}(d_1, d_2) = 1.0 - \frac{|Ind(d_2) - Ind(d_1)|}{9} \tag{8.8}$$

$$Ind(d) = Indirectness(d) + Strength(d) \tag{8.9}$$

8.3 Computing the similarity of conceptual configurations

As we just saw, we sometimes have to compute the similarity of whole conceptual structures, rather than of mere primitive concepts. For instance, we might need to decide how similar an extremely severe criticism is to a moderately severe criticism, or to just severe criticism, or to mere criticism. Or, we might need to decide how similar official authority is to peremptory authority, or how similar arbitrary power is to peremptory authority (where arbitrariness is a kind of peremptoriness, and authority is a kind of power). Computing this type of similarity is clearly different than but related to the problem of computing the similarity of primitive concepts (or words). We have to consider not only the content but also the structure of the configurations.

To make this problem more concrete, consider the pairs of conceptual structures shown in figure 8.3. Despite the differences between each of these pairs of concepts, they are nevertheless quite similar; we'd like to know how similar. Consider the following general cases (and suggestions for how one might compute a similarity score in each):

$$\begin{aligned}
\text{If}\quad d_1 &= (lex_1 \text{ low formality}) \quad \text{and} \\
d_2 &= (lex_2 \text{ medium formality}) \\
\text{then}\quad Sim(d_1,d_2) &= 0.5
\end{aligned}$$

$$\begin{aligned}
\text{If}\quad d_1 &= (lex_1 \text{ always strong favourable V1}) \quad \text{and} \\
d_2 &= (lex_2 \text{ usually medium pejorative V1}) \\
\text{then}\quad Sim(d_1,d_2) &= S_{freq}(d_1,d_2) \times S_{att}(d_1,d_2) \\
&= 0.75 \times 0.167 \\
&= 0.125
\end{aligned}$$

$$\begin{aligned}
\text{If}\quad d_1 &= (lex_1 \text{ always strong implication P1}) \quad \text{and} \\
d_2 &= (lex_2 \text{ sometimes weak suggestion P1}) \\
\text{then}\quad Sim(d_1,d_2) &= S_{freq}(d_1,d_2) \times S_{ind}(d_1,d_2) \times S_{con}(d_1,d_2) \\
&= 0.5 \times 0.375 \times 1.0 \\
&= 0.1875
\end{aligned}$$

$$\begin{aligned}
\text{If}\quad d_1 &= (lex_1 \text{ always medium implication P1}) \quad \text{and} \\
d_2 &= (lex_2 \text{ never medium implication P1}) \\
\text{then}\quad Sim(d_1,d_2) &= S_{freq}(d_1,d_2) \times S_{ind}(d_1,d_2) \times S_{con}(d_1,d_2) \\
&= 0.0 \times 1.0 \times 1.0 \\
&= 0.0
\end{aligned}$$

Figure 8.2: Examples of computing the similarity of lexical distinctions.

1.	'high	'medium
2.	(P3 Criticism (ACTEE (P3-0 John)) (ATTRIBUTE (P3-1 Severity (DEGREE 'high))))	(Q3 Criticism (ACTEE (Q3-0 John)) (ATTRIBUTE (Q3-1 Severity (DEGREE 'medium))))
3.	(P3 Criticism (ACTEE (P3-0 John)) (ATTRIBUTE (P3-1 Severity (DEGREE 'high))))	(Q3 Criticism (ACTEE (Q3-0 John)) (ATTRIBUTE (Q3-1 Severity)))
4.	(P3 Criticism (ACTEE (P3-0 John)) (ATTRIBUTE (P3-1 Severity (DEGREE 'high))))	(Q3 Criticism (ACTEE (Q3-0 John)))
5.	(P1 Authority)	(Q1 Power)
6.	(P1 Authority (ATTRIBUTE (P1-1 Peremptory)))	(Q1 Authority (ATTRIBUTE (Q1-1 Official)))
7.	(P1 Authority (ATTRIBUTE (P1-1 Peremptory)))	(Q1 Power (ATTRIBUTE (Q1-1 Arbitrary)))

Figure 8.3: Seven pairs of similar conceptual structures.

1. If the concepts are primitive (e.g., pair 5 in figure 8.3), then we could use an existing similarity measure (Budanitsky, forthcoming 1999). (The concepts might be related by subsumption, or to the same parent concept, or in some other way).

2. If the concepts are fuzzy (e.g., pair 1), or involve numeric values, then we could use a fuzzy-set conformity function to determine the similarity of the concepts (see section 7.3.6).

3. If the concepts have composite structures (e.g., pairs 2, 3, 4, 6, and 7), then the problem is much more complex. Since each structure has a root concept and a set of sub-structures, we could compute an overall similarity score by recursively computing similarity scores of pairs of sub-structures (with the above two cases as base cases). But to do this, we must first be able to determine which pairs of sub-structures should be compared. For instance, in pair 2, it's obvious that we would not want to compute the similarity of the sub-structures `(P3-1 Severity (DEGREE 'high))` and `(Q3-0 John)` but rather of `(P3-1 Severity (DEGREE 'high))` and `(Q3-1 Severity (DEGREE 'medium))`. But it's not always obvious; in pair 6, for instance, should we compute the similarity of `(P1-1 Peremptory)` and `(Q1-1 Official)`? It might also happen that one of the root concepts has no sub-structure that corresponds to any of those of the other root concept, as in pairs 3 and 4; but this can be difficult to determine, because we don't expect sub-structures to match exactly. Finally, some sub-structures would seem to be more significant than others. For instance, in pair 2, having a different actee would be more significant than having a different degree of criticism. So, we are left with following problem: it seems that before we can compare two sub-structures (to determine their similarity), we have to first determine which sub-structures are similar enough to be compared.

These cases are of course a gross simplification, since we have assumed that concepts are represented in canonical-form graph structures that can be fairly easily matched. This might not be the case: configurations of quite different structure may still represent similar concepts. We are not aware of any research on the general problem of computing the similarity of arbitrary conceptual structures, but some related work has been done in the area of description logics. Cohen, Borgida and Hirsh (1992), for example, formalize a "least common subsumer" (LCS) operation, which returns the largest set of commonalities between two descriptions. For instance, the LCS of the concepts of pair 7 is `(X1 Power (ATTRIBUTE (X1-1 Peremptory)))`. And Baader, Borgida and McGuinness (1998) give preliminary results of a matching operator that matches variables in one (arbitrary) description to concepts in another.

However, our problem is actually more specific than this general problem, because the conceptual structures that we would like to compare are not arbitrary but represent values on particular dimensions of variation within a cluster. Thus, we can assume that any pair of concepts will have similar structures up to a certain point, because they will both be extensions of the same peripheral concept. We might also be able to assume that the sub-structures that make up an extension will each be related to the peripheral concept by a unique relation (e.g., that there will be at most one sub-structure that is an `ATTRIBUTE`, at most one `DEGREE`, and so on). Finally, a similarity measure should be able to take advantage of the constraints on the fillers of peripheral concepts; for example, since the dimension of `Authority` can take only `ATTRIBUTE`s of type `Official` or `Peremptory`, we might be able to use the ontological relationships between officialness and peremptoriness in computing similarity.

So, from this preliminary characterization of the problem, one can see that developing a full solution will require much more research, which is beyond the scope of this thesis. We developed, instead, a very naive conceptual-similarity function, which relies on some of the assumptions discussed above. The function can be improved as future theoretical results become available.

The function computes similarity by simultaneously traversing a pair of conceptual structures. Instead of looking for an exact match between pairs of sub-structures, it recursively computes the similarity of two sub-structures if each is related to its root concept by the same relation. For instance, in pair 3 in figure 8.3, the sub-structures `(P3-1 Severity (DEGREE 'high))` and `(P3-1 Severity)` are compared because they are both `ATTRIBUTE`s of their respective root concepts. The similarity function then combines the scores of the pairs of sub-structures with the similarity of the two root nodes (which are always primitive or fuzzy concepts). The scores for the sub-structures are given less weight in the calculation, because, intuitively, the deeper one goes in a conceptual structure, the less significant any differences become. Any leftover unmatched sub-structures are considered to have zero similarity (to their non-existent partner). Hence, the degree of similarity of the conceptual structures t_1 and t_2 is

$$S(t_1,t_2) = \begin{cases} \mathit{Similarity}(\mathit{root}(t_1),\mathit{root}(t_2)) & \text{if } N(t_1,t_2)=0 \\ \alpha \mathit{Similarity}(\mathit{root}(t_1),\mathit{root}(t_2)) + \beta \frac{1}{N(t_1,t_2)} \sum_{s_1,s_2} S(s_1,s_2) & \text{otherwise} \end{cases} \tag{8.10}$$

where $N(t_1,t_2)$ is the sum of the number of matched sub-structures and the number of unmatched sub-structures in t_1 and t_2, s_1 and s_2 are any pair of matching sub-structures of t_1 and t_2, *Similarity* is a function that returns the similarity of primitive concepts, and α and β are weighting factors. We set $\alpha = \beta = 0.5$, so the whole set of sub-structures is weighted the same as the root concepts are weighted.

Given two primitive concepts c_1 and c_2, $\mathit{Similarity}(c_1,c_2)$ is computed as follows:

- If c_1 and c_2 both fill a `DEGREE` role, then they are fuzzy sets or numeric values (as constrained by the ontology). So, we use the fuzzy-set conformity function, defined in section 7.3.6, to compute their similarity.

- If c_1 and c_2 are primitive concepts defined in the ontology, then regardless of what role they fill, the function returns 1.0 if either one subsumes the other (and 0.0 otherwise), But really, the function should take into account the role they fill (i.e., whether they are `ATTRIBUTES`s, `ACTOR`s, etc.), and should use an existing similarity measure (Budanitsky, forthcoming 1999).

- Otherwise, it returns 0.0.

Examples

For each of the pairs of structures shown in figure 8.3, taking the left-hand and right-hand structures as t_1 and t_2, respectively, the similarities are:

1. $S(t_1,t_2) = 0.6$
2. $S(t_1,t_2) = 0.95$
3. $S(t_1,t_2) = 0.875$
4. $S(t_1,t_2) = 0.75$
5. $S(t_1,t_2) = 1.0$

6. $S(t_1, t_2) = 1.0$
7. $S(t_1, t_2) = 1.0$

To expand on the second pair, the function used the following calculations to arrive at the value of 0.95:

$$S(t_1, t_2) = \alpha c_1 + \beta \tfrac{1}{2}(c_2 + c_3) = 0.95$$

where

$$\begin{aligned}
c_1 &= \mathit{Similarity}(\texttt{Criticism}, \texttt{Criticism}) = 1.0 \\
c_2 &= \mathit{Similarity}(\texttt{John}, \texttt{John}) = 1.0 \\
c_3 &= \alpha c_4 + \beta c_5 = 0.8 \\
c_4 &= \mathit{Similarity}(\texttt{Severity}, \texttt{Severity}) = 1.0 \\
c_5 &= \mathit{Conformity}(\texttt{high}, \texttt{medium}) = 0.6
\end{aligned}$$

On the fourth pair, the function used the following calculations to arrive at 0.75:

$$S(t_1, t_2) = \alpha c_1 + \beta \tfrac{1}{2}(c_2 + 0) = 0.75$$

where

$$\begin{aligned}
c_1 &= \mathit{Similarity}(\texttt{Criticism}, \texttt{Criticism}) = 1.0 \\
c_2 &= \mathit{Similarity}(\texttt{John}, \texttt{John}) = 1.0
\end{aligned}$$

And on the seventh pair, the function used the following calculations to arrive at 1.0:

$$S(t_1, t_2) = \alpha c_1 + \beta c_2 = 1.0$$

where

$$\begin{aligned}
c_1 &= \mathit{Similarity}(\texttt{Authority}, \texttt{Power}) = 1.0 \\
c_2 &= \mathit{Similarity}(\texttt{Peremptory}, \texttt{Arbitrary}) = 1.0
\end{aligned}$$

Chapter 9

Automatic lexical choice

The clustered model of lexical knowledge that we have developed in chapter 7 can be applied in any natural language system that requires a knowledge of fine-grained lexical semantics. In this chapter, we explore a major application of the model: *automatic lexical choice* in text generation.

Lexical choice, as we see it, is more than a problem of mapping from concepts to words; it is a problem of meeting or satisfying a large set of possibly conflicting preferences to express certain nuances in certain ways, to establish the right style, and to observe collocational and syntactic constraints through the appropriate selection of words. So, lexical choice—genuine lexical choice—is about making choices between options, rather than merely finding the words for concepts, as is the case in many past text generation systems.

In fact, this kind of sophisticated lexical choice is the central task in text generation (or, at least, sentence generation), because it interacts with almost every other task involved (see chapter 4). Therefore, it would be difficult to develop and implement a model for sophisticated lexical choice without also building a complete sentence generation system. Fortunately, such systems were the outcome of earlier attempts at developing sophisticated lexical choice—inevitably, I believe, since lexical choice had to be sidelined until models and systems for basic sentence planning were developed (Elhadad, McKeown and Robin, 1997; Stede, 1999). But now that we have these systems, we can select one of them to provide a basic processing infrastructure and focus directly on sophisticated lexical choice.

This chapter describes our system, called **I-Saurus**, which is a modified version of Stede's MOOSE system. We start with an overview of I-Saurus, followed by descriptions of its input and output structures. Then we discuss the sentence-planning algorithm and, finally, the lexical-choice algorithm.

9.1 I-Saurus

We have developed an algorithm for automatic lexical choice and implemented it in our prototype system, **I-Saurus**. I-Saurus is a sentence generation system, but its forte is choosing from among a cluster of near-synonyms the one that best satisfies a set of input preferences for conveying semantic, stylistic, and attitudinal meaning. I-Saurus is based on MOOSE (Stede, 1999), from which we took the basic sentence generation components for producing paraphrases (including functions to match word denotations to input representations and to build up sentence plans). To incorporate the new clustered model of the lexicon, substantial modifications were required to some of MOOSE's components and to the overall architecture of the system. So,

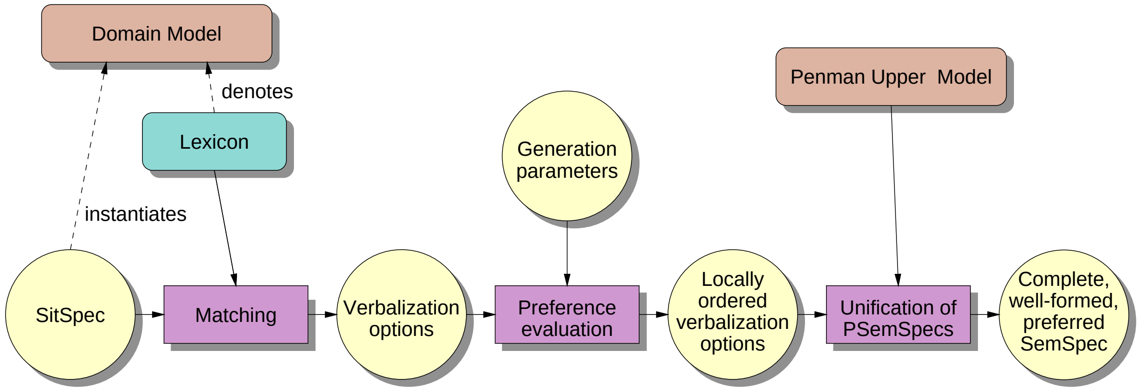


Figure 9.1: System architecture of MOOSE. (Adapted from Stede (1999, p. 148).)

before we describe I-Saurus, we must give a brief overview of MOOSE.

9.1.1 Overview of MOOSE

Figure 9.1 shows the architecture of the MOOSE system. The input to MOOSE is a representation of meaning (on the conceptual–semantic level) called a *SitSpec*. A SitSpec is a graph of instantiated domain concepts linked by domain relations that specifies a situation to be lexicalized. For example, we could input the following SitSpec to MOOSE:

```
(say1  (SAYER john1)
       (SAYING lie1_F))
```

This represents the situation of a person (`john1`) saying (`say1`) a 'lie' (`lie1`), which could be realized as "John tells a lie". The `_F` feature on `lie1` indicates a preference to foreground, or make salient, the lie in the output, which could be realized as "A lie was told by John". From a SitSpec, MOOSE builds and outputs a *SemSpec*—a lexicalized sentence plan in Sentence Plan Language (SPL), which serves as input to Penman. The process involves three sources of knowledge: a domain model (or ontology), a lexicon in which lexical entries both denote configurations of domain concepts and link to SemSpec fragments (or PSemSpecs), and the Penman Upper Model.

Processing is done in three stages:[1]

Matching A set of *verbalization options* is determined by finding the lexical items that cover parts of the SitSpec. Each SitSpec node is linked to the options whose denotations match a subgraph of the SitSpec rooted at the node.

Figure 9.2 depicts the state of the system after matching has been completed on the SitSpec, which is shown here in graphical form. Each node of the SitSpec is linked to its verbalization options. For instance, the `say1` node is linked to the options for *tell*, because *tell* covers the one node, and *lie* (the verb) and *prevaricate*, because they each cover both the nodes `say1` and `lie1`.

[1] We have omitted from the following description a major component of the system that applies verb alternations and extensions, because it is not relevant to the present discussion.

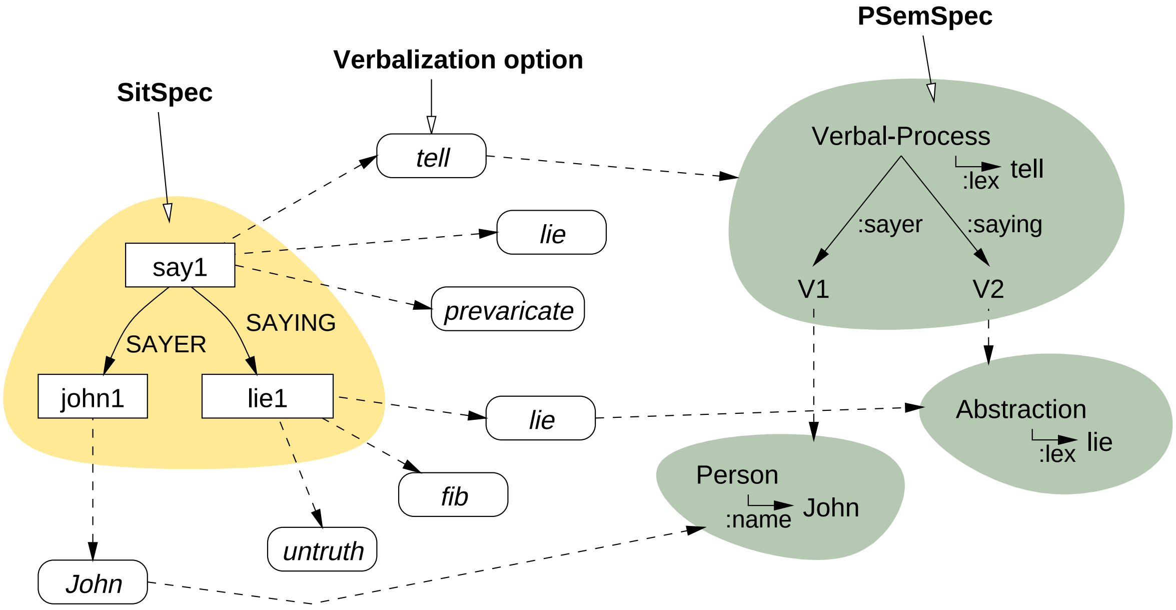


Figure 9.2: The state of MOOSE after the matching stage on the SitSpec for "John tells a lie". In the SitSpec, solid arrows indicate relations between instances of domain concepts. In the PSemSpecs, solid arrows indicate relations between Upper Model concepts. Dashed arrows link SitSpec nodes to their verbalization options, and from there to PSemSpecs.

Since a denotation is a necessary applicability condition for a word, a denotation matches a SitSpec subgraph if it subsumes (i.e., is more general than) the concept of which the subgraph is an instance. The matching algorithm, a standard recursive graph-matching algorithm, is simple and efficient because it relies on the tree-like structure of denotations and SitSpecs. A denotation and SitSpec subgraph match if

- the root node of the denotation is a superconcept of (or the same as) the type of the root node of the SitSpec,
- every relation of the root node of the denotation matches a relation of the root node of the SitSpec (that is, the denotation must not have any extra relations, but it may have fewer relations), and
- the fillers of the relations also match, recursively.

Preference evaluation Using the supplied generation parameters, the system ranks the list of verbalization options associated with each SitSpec node. For instance, in our SitSpec, a parameter asks that the instance of the lie (`lie1`) be foregrounded, and one way to do this is to use the passive form of the verb *tell*, so this verbalization option would be ranked higher than the active option. Stede made few claims about how preference evaluation was to take place; his main idea was that all of the options be made ready and somehow ordered according to user preferences.

Unification of PSemSpecs The final stage is the core process of the system. It attempts to find a subset of the verbalization options such that (1) the partial SemSpecs associated with

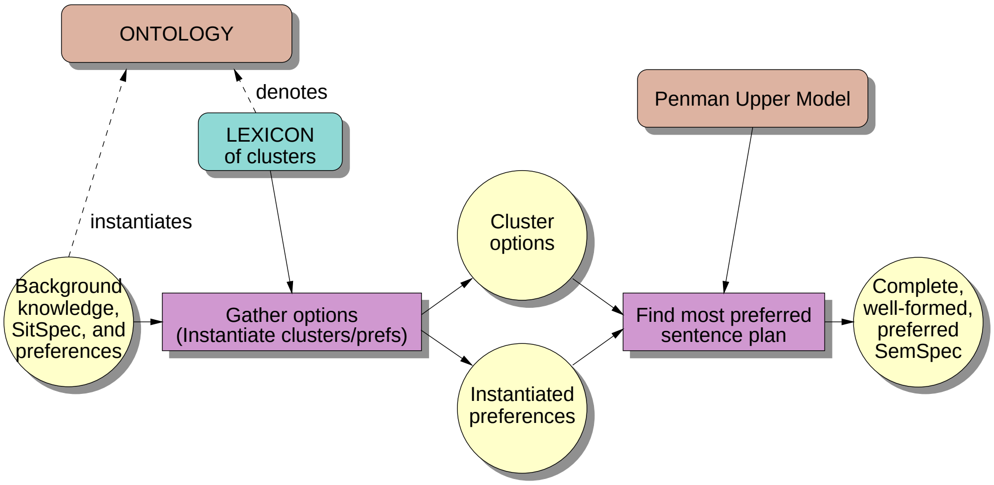

ONTOLOGY
denotes
LEXICON of clusters
instantiates
Penman Upper Model
Cluster options
Background knowledge, SitSpec, and preferences
Gather options (Instantiate clusters/prefs)
Find most preferred sentence plan
Complete, well-formed, preferred SemSpec
Instantiated preferences

Figure 9.3: System architecture of I-Saurus.

> the options can be unified into a well-formed SemSpec (according to the Penman Upper Model); (2) the SemSpec is complete, or the denotations of the chosen options completely cover the SitSpec with no redundancies; and (3) the SemSpec is preferred. In figure 9.2, for instance, the system can choose the options *John*, *tell*, and *lie*, forming the complete SemSpec shown on the right (once the variables `V1` and `V2` are replaced with the PSemSpecs to which they point). Or, the system could choose *John* and *lie* (the verb), which also completely cover the SitSpec.
>
> We will not go into the details of the algorithm here, since we give our modified version of it later, but we will give an example. In figure 9.2, after choosing the verbalization option for *tell*, the PSemSpec linked to it becomes the basis for the final SemSpec. The external variables `V1` and `V2` of the PSemSpec refer to the SitSpec nodes `john1` and `lie1`, so SemSpecs are built for their chosen verbalization options (*John* and *lie*). Finally, all three PSemSpecs are unified into a complete SemSpec.

As even this simple example shows, the MOOSE system can produce different paraphrases of a SitSpec both by covering the SitSpec in different ways (with words that have structurally different denotations) and by choosing different synonyms (that have the same denotation). However, as it stands, it cannot actually decide between the different paraphrases, because the decision-making criteria and the required lexical knowledge have not been formalized in the system. But clearly, MOOSE is sufficiently sophisticated that we can formalize criteria (i.e., preferences), and incorporate into it the clustered model of lexical knowledge.

9.1.2 Overview of I-Saurus

Figure 9.3 shows the overall architecture of I-Saurus. The input is a situation, which consists of background knowledge, a MOOSE-style SitSpec, and a set of preferences. The SitSpec represents a constraint on what must be lexicalized (out of the background knowledge), and the preferences describe the semantic, stylistic, and attitudinal aspects that the user would like the system to attempt to convey. The output is a complete well-formed preferred SemSpec (or a ranked set of complete well-formed SemSpecs), as in MOOSE. The representational formalisms for the input and output are discussed in sections 9.2 and 9.3 below.

Processing occurs in two stages:

Gather options Like MOOSE, I-Saurus first gathers all of the options for choice, but the major difference in I-Saurus is that we have replaced verbalization options with *cluster options.* A cluster option is an 'instantiation' of a lexical cluster. During processing, each cluster option maintains covering information and variable bindings, and runs a lexical choice process. A cluster is instantiated if its core denotation matches a subgraph of the input SitSpec (using MOOSE's matching algorithm). Once instantiated, a link is made to the cluster option from the SitSpec node that its core denotation matches, and its variables are bound back to their corresponding SitSpec nodes. For example, figure 9.4 depicts the state of I-Saurus after the first stage of processing on the input for "John tells a lie" (and its paraphrases). Four clusters have been instantiated: two, `say_C` and `tell-a-lie_C`, match subgraphs of the SitSpec rooted at `say1`; one, `untruth_C`, matches just the `lie1` node; and one matches `john1` (see appendix B for the representations of all of these clusters). The `say_C` cluster binds its variables `ROOT`, `V1`, and `V2` to the `say1`, `john1`, and `lie1` nodes, respectively (as indicated by dotted arrows). The other clusters bind their variables in a similar fashion, but this is not shown in the diagram. The `untruth_C` cluster is shown with its core denotation unfolded. Although the variable `V1` is depicted as being bound to `john1`, it would actually be bound only if it was necessary to do so (during processing in the second stage).

The first stage also instantiates the input preferences, but no preferences are shown in the example. (Preferences will be discussed in section 9.2.3 below.) The process would link each instantiated preference to the cluster options that could potentially satisfy the preference, if the appropriate near-synonym were chosen. For instance, a preference to imply that the lie is insignificant, in the above example, would be linked to both `tell-a-lie_C` and `untruth_C`, because the former has the verb *fib* and the latter the noun *fib*.

Our implementation notes, in section 9.6, give more details on the first stage of processing.

Find most preferred sentence plan The second stage is a combination of MOOSE's second and third stages, but the main difference is a result of our use of cluster options instead of verbalization options. Now, two different mutually constraining types of decision must be made:

- Choosing from several competing cluster options.
- Choosing a near-synonym from a cluster option.

We believe that it is important to separate the processes for making these two types of decision, even though they must interact, because of their different choice criteria and effects. The former type involves choosing between options of differing coarse-grained semantic content and resulting syntactic structure—clusters have different core denotations after all. Here, issues of syntactic and semantic style are involved, as one can choose among different syntactic structures, or choose how the semantic content is to be incorporated. On the other hand, the latter type of decision involves options that might have subtle semantic and stylistic differences but result in the same syntactic structure (though collocational structure can vary).

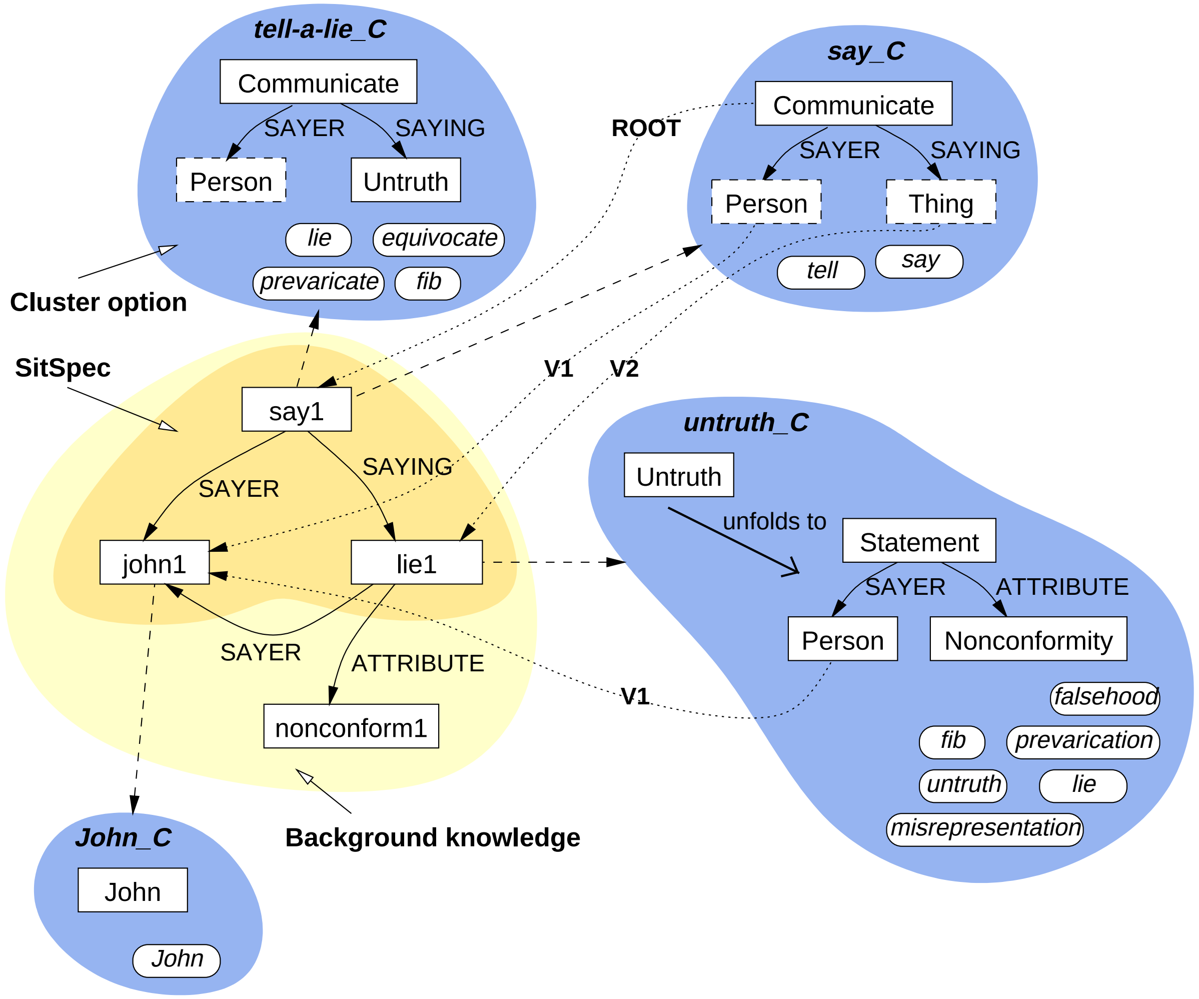


Figure 9.4: The state of I-Saurus after the first stage of processing on the input for "John tells a lie". Four clusters have been instantiated; they are shown each with its core denotation and set of near-synonyms. Solid arrows in the SitSpec and background knowledge indicate relations between instances of concepts. Solid arrows in the cluster options link concepts in the core denotations. Dashed arrows link SitSpec nodes to the cluster options that cover subtrees rooted at the nodes. Dotted arrows represent variable bindings, and link concepts in the core denotations back to their corresponding SitSpec nodes. (Bindings are shown for only `say_C` and one variable of `untruth_C`.)

In other words, lexical choice is a two-tiered process that must find both the appropriate set of cluster options and the appropriate set of lexical items (one from each chosen cluster option) whose PSemSpecs can be unified into a complete well-formed SemSpec. Of course, many possible SemSpecs can usually be generated, but the real problem is to find the combination of cluster options and lexical items that *globally* satisfy as many of the input preferences as possible. For instance, in figure 9.4, we could choose the `tell-a-lie_C` cluster and the `John_C` cluster, which fully cover the SitSpec, and then choose the words *John* and *lie* to come up with "John lies" (or we could choose *John* and *prevaricate* for "John prevaricates"). We could also choose the `say_C`, `untruth_C`, and `John_C` clusters, and then the words *tell*, *fib*, and *John*, to end up with "John tells a fib". These alternatives—there are many others—are different in structure, meaning, and style. This problem of decision-making is theoretically and computationally significant; we'll return to the issues involved after describing the formalisms for the input and output.

9.2 Input: Constraints and preferences

Past systems for lexical choice were not complex enough to need much more than a few concepts as input, because they had to make sure only that the words chosen were applicable to the situation, that is, that their denotations matched (exactly) the input. But now that we have a model of lexical knowledge that represents fine-grained aspects of meaning, the lexical choice process will necessarily be more complex, so, in turn, the input to the process must also be more complex. Otherwise, we couldn't take advantage of the fine-grained knowledge. But with so many possibilities and options, choosing from among them will necessarily involve not only degrees of satisfying various criteria, but also tradeoffs between different criteria. Thus, we must formalize the criteria not as hard *constraints* but as *preferences*. However, we need some constraints to ensure that the basic desired meaning is nevertheless accurately conveyed (see (Stede, 1995)).

So, the input to I-Saurus is a specification of a *situation* to be lexicalized, which consists of: background knowledge, a specification of the aspects of the situation that must be lexicalized, and a set of preferences about how peripheral aspects of the situation should be conveyed.

9.2.1 Background knowledge

The background knowledge of a situation is a set of assertions about some entities and their relationships in the world. It is the knowledge against which the other two components of the situation are to be understood, but none of its aspects are not meant to be necessarily lexicalized in the output of the system. In a typical knowledge base, many facts about the participants of a situation would be known, only some of which would be relevant to a speaker's goals in lexicalizing the situation. For example, a speaker might believe, for a situation involving a person's error, that the person was negligent in causing the error, but might not want to imply this in any way in lexicalizing the situation; the speaker would then exclude this fact (or belief) from the background knowledge of the situation. Thus, the background knowledge is a subset of the whole knowledge base, containing only the facts relevant to lexicalization. This formulation is a departure from MOOSE, where the input consists of a specification of all and only the facts of a knowledge base that must be lexicalized. We found that the distinction between background knowledge and knowledge-to-be-lexicalized was necessary for the representation and processing of preferences.

sitspec	::=	(*node relation*∗)
node	::=	⟨any instance of a concept defined in the ontology⟩
relation	::=	(*role filler*)
role	::=	⟨any relation defined in the ontology⟩
filler	::=	*node* | ’ *value* | *sitspec*
value	::=	⟨0..1⟩ | ⟨a fuzzy set on 0..1⟩

Figure 9.5: Syntax of a specification of the aspects of a situation to be lexicalized (a simplified MOOSE-style SitSpec).

We formalize a situation as a set of instances (of the concepts defined in the ontology) and assertions about those instances. For example, the following Loom assertions:

```
(tell (John john1))
(tell (Nonconformity nonconform1))
(tell (Statement lie1)
      (SAYER lie1 john1)
      (ATTRIBUTE lie1 nonconform1))
(tell (Say say1) (SAYER say1 john1) (SAYING say1 lie1))
(tell (Mislead mislead1))
(tell (Intend intend1) (ACTOR intend1 john1) (ACTEE intend1 mislead1))
```

state, for instance, that `john1` is an instance of the concept `John` (i.e., a `Person` whose name is “John”), and that `john1` is the sayer of `lie1` (an instance of `Statement`), which has the attribute of `nonconform1` (an instance of `Nonconformity`). Using its algorithm for computing subsumption, Loom can figure out that `lie1` is an instance of `Untruth`.

9.2.2 Constraints

The aspects of the input situation that must be conveyed (i.e., the constraints on lexical choice) are represented as a simplified MOOSE-style SitSpec, that is, as a graph of instances linked by relations. Figure 9.5 shows our simplified SitSpec syntax. It is similar to the syntax for core denotations, since, after all, core denotations are matched to SitSpecs. Although a SitSpec is a graph of nodes, it is represented as a tree with a root node and subtrees for each role filler. For example, the SitSpec of figure 9.4 is represented as follows:

```
(say1
   (ACTOR john1)
   (ACTEE lie1))
```

A more complicated SitSpec, which involves co-reference, using instances of a situation not defined here, is the following:

```
(order1
   (SAYER john1)
   (SAYEE mary1)
   (SAYING (perform1
```

Type of preference	Description
Denotational / semantic	Express a concept in a certain way (e.g., with a certain degree of indirectness).
Expressive / attitudinal	Express an attitude towards an entity.
Stylistic	Establish a certain style.
Emphatic	Emphasize an aspect of the situation.
Salience	Foreground or background an entity.
Typical usage	Use typical words and structures.
Specificity	Use more specific or more general terms.
Dialectal	Use terms from a dialect or sublanguage.
Lexical	Use or don't use a particular word.
Morphological	Use a term with a particular morphology (e.g., one that rhymes).

Table 9.1: Types of preference pertaining to lexical choice.

```
        (ACTOR mary1)
        (ACTEE (accompany1
                  (ACTOR mary1)
                  (CO-ACTOR john1))))))
```

This SitSpec could be lexicalized as "John ordered Mary to accompany him."

9.2.3 Preferences

The main difference between a constraint and a preference is that a preference can be satisfied to different degrees, or even not at all, depending on the decisions that are made during sentence planning. A preference can be satisfied by a single decision, or by a group of decisions.[2] And, because conflicts and tradeoffs might arise in the satisfaction of several preferences at once, each preference has a user-specified importance factor.

There are many types of preference that one could have when generating a sentence, only some of which pertain to lexical choice. Table 9.1 lists several types of preference, which essentially mirror the types of lexical variation that we classified in chapter 3. Of course, many of these types of preference don't pertain solely to lexical choice; many could in fact be satisfied by any sort of decision made in sentence planning. In our system, we formalized only the first three types listed, namely, denotational (or semantic), expressive, and stylistic preference. But the other types could also be formalized in the system—some would be relatively easy to incorporate (e.g., lexical preferences), others would require more research.

On a different dimension, preferences are of two types: local and global. A *local preference* is one that involves a particular entity or aspect of the situation to be lexicalized. Usually, a local preference will be related to one or more particular clusters, because only those clusters can po-

[2] Thus, a preference is like a *floating constraint* (Elhadad, McKeown and Robin, 1997) in that it can be satisfied by different types of decision in sentence-planning, but unlike a floating constraint, it can be satisfied to different degrees (see section 4.2.5).

preference	::=	*semantic-pref* | *expressive-pref* | *stylistic-pref*
semantic-pref	::=	(*[importance] method sitspec*)
expressive-pref	::=	(*[importance] stance node*)
stylistic-pref	::=	(*[importance] style-value stylistic-dimension*)
importance	::=	⟨0..1⟩
method	::=	*suggest* | *imply* | *denote*
stance	::=	*favour* | *neutral* | *disfavour*
sitspec	::=	⟨defined in figure 9.5⟩
node	::=	⟨defined in figure 9.5⟩
style-value	::=	⟨defined in figure 7.11⟩
stylistic-dimension	::=	⟨defined in figure 7.11⟩

Figure 9.6: Syntax of preferences.

tentially satisfy the preference. So, choosing one right word is usually enough to satisfy a local preference. For instance, one can satisfy a preference to express a pejorative attitude towards someone by making one word choice (e.g., by choosing *skinny* instead of *slim*). Semantic and expressive preferences are local. *Global preferences*, on the other hand, do not directly involve particular entities of the situation, and can be satisfied by potentially any cluster. In fact, they must usually be satisfied collectively and continuously by several word choices. Stylistic preferences are global, because one cannot establish, say, a formal style, by choosing merely one formal word—many choices, including syntactic choices, are involved. (Hovy (1988) discusses the same distinction using a terminology of goal satisfaction in planning.)

We formalize preferences as shown in figure 9.6. (Importance defaults to 1.0.) *Semantic preferences* include the preferred method to use, either to “suggest”, “imply”, or “denote”, and a specification of the parts of the situation that should be conveyed. The specification is made in the same formalism used for SitSpecs, which is important, because there is no theoretical difference in the nature of a preferred meaning to a ‘constrained’ meaning. The only difference is in how and why they are conveyed. Also, for it to be possible to satisfy a semantic preference, the specification must represent a ‘value’ on a dimension (which is itself a configuration of concepts), just as semantic distinctions do.

Expressive preferences include the stance (or attitude) that the speaker wishes to take, either to “favour”, be “neutral” to, or “disfavour” an entity of the situation, and the entity’s instance in the situation. Finally, *stylistic preferences* specify a stylistic dimension and a desired value on the dimension.

The following examples show several preferences related to our running example of John and his lie:

;; suggest that John is trying to save face

```
(suggest (saveface1 (ACTOR john1)))
```

;; imply that the lie is insignificant

```
(imply (significant1 (ATTRIBUTE-OF lie1) (DEGREE ’low)))
```

;; imply that John has a deliberate intent to mislead

```
(imply (intend1 (ACTOR john1) (ACTEE mislead1)))
```

;; imply that John is criticized, with half importance

```
(0.5 imply (criticism1 (ACTEE john1)))

;; imply varying degrees of severity of criticism
(imply (criticism1 (ACTEE john1)
                   (ATTRIBUTE (severe1 (DEGREE ’0.7)))))
(imply (criticism1 (ACTEE john1)
                   (ATTRIBUTE (severe1 (DEGREE ’high)))))
(imply (criticism1 (ACTEE john1)
                   (ATTRIBUTE (severe1 (DEGREE
                                        ’(OR medium high))))))

;; ask to express a pejorative attitude towards John
(disfavour john1)

;; establish an informal style
(low formality)
```

9.3 Output: A sentence plan

The output of I-Saurus is a specification of the meaning of a sentence on the syntactic–semantic level, that is, a MOOSE SemSpec. A SemSpec is a sentence plan in SPL (i.e., a structure of concepts and relations defined in the Penman Upper Model), from which Penman can generate a sentence. Unlike a normal Penman sentence plan, a SemSpec is fully lexicalized (i.e., all open-class lexical items have already been chosen by our system), so Penman isn’t required to make any lexical choices, except for closed-class words such as prepositions and determiners. We used MOOSE’s SemSpec formalism, so we will not repeat it here.

For example, I-Saurus outputs the following SemSpec, among others, for our example:

```
(MAKE1 / Non-Addressee-Oriented-Verbal-Process
       :lex tell
       :sayer (JOHN1 / Person
                     :name John)
       :saying (LIE1 / Abstraction
                     :lex lie))
```

which Penman realizes as:

```
"John tells a lie."
```

9.4 The sentence-planning algorithm

In the general case, given a set of input preferences (of many different types), a sentence planner will make a large number of decisions (also of different types), each of which has the potential to satisfy any, some, or all of the input preferences. As we said above, it is unlikely that any particular set of preferences can all be satisfied simultaneously at once, so some kind of conflict resolution strategy is required in order to manage the decision-making task (Hovy, 1988;

Nirenburg, Lesser and Nyberg, 1989; Wanner and Hovy, 1996). It is not within the scope of this thesis to further develop solutions to this general problem. Instead, in this section we will (1) discuss the new issues that arise in managing the interactions between lexical choices, (2) suggest a general framework for sentence planning that has the potential to manage these interactions, and (3) describe our sentence-planning algorithm, which is a modification of an existing algorithm.

So, consider an input to our system that consists of background knowledge, a SitSpec, and a set of preferences of the three types that we defined above. Once the system has gathered all of the cluster options for the input, it must find (and build) the sentence plan (i.e., SemSpec) that best satisfies as many of the input preferences as possible. As we mentioned above, the system can make two types of decision, between cluster options and between near-synonyms. Now, if none of the available options or near-synonyms will satisfy, to any degree, a particular preference, then that preference is trivially impossible to satisfy. But even when there are options that will satisfy a preference, it still might be impossible to satisfy the preference, because various types of conflicts can arise in trying to satisfy several preferences at once.

For instance, in choosing a particular cluster option in order to satisfy one preference, we might not be able to satisfy another preference that is only satisfiable by a different competing cluster option. (E.g., one might choose the `err_C` cluster because of the simplicity or directness of its syntax: "John erred", but we would not be able to simultaneously also satisfy a preference for implying a misconception by choosing, say, *mistake* from the `error_C` cluster: "John made a mistake".) Similar tradeoffs occur when choosing among the near-synonyms of the same cluster, but we will cover this case in section 9.5 below.

Two separate simultaneous choices might also conflict in their satisfaction of a single preference. That is, the preference might be satisfied by one choice and negatively satisfied by another choice. For instance, it might happen that one word is chosen in order to express a favourable attitude towards a participant of the situation while another word is chosen that inadvertently expresses a pejorative attitude towards the same person, if that second word is chosen in order to satisfy some other preference.

Similarly, in the case of a global preference, it could be difficult to make enough choices to sufficiently satisfy the preference. For instance, if one inputs a preference for formal style, what happens if three formal words and one informal word are chosen? Does a formal style win out? It might depend on other aspects of the words (e.g., other nuances), or on their positions in the sentence. Generally, it can be difficult to determine if a set of choices pertaining to the same preference will amplify, dampen, or cancel the overall effect.

Clearly the interactions are complex and context-dependent, so, finding the best set of options could involve a lengthy search process. There are two separate issues here. The first is theoretical: to analyze and formalize how the decisions interact. And the second is practical: to develop a computationally efficient process for searching for the best set of options. Both are hard (and intriguing) open problems for which we can give only preliminary solutions here, pending future research. But before we get to these solutions, we must first discuss two fundamental design decisions that are required for any solution to be effective.

First, we will model a preference as an object that can be reasoned about and that maintains information about its satisfaction. That way, once the interaction of decisions on the satisfaction of preferences is better understood, it can be modelled as a reasoning process involving preferences. Second, we will model numerically the degree to which a word satisfies a preference. Not only will this make it easy and efficient to compare and rank different options in order to find the best set, but it will make it possible to account for tradeoffs in the satisfaction of preferences.

```
Build-Sentence-Plan(node)
    (1) $c \leftarrow$ Next-Best-Cluster-Option (node)
    if we've tried all the options then return "fail"
    (2) $w \leftarrow$ Next-Best-Near-Synonym($c$)
    if we've tried all the near-synonyms then backtrack to (1)
    $p \leftarrow$ PSemSpec of $w$
    if $p$ has no external variables
        then return $p$
        else for each external variable $v$ in $p$
                $s \leftarrow$ Build-Sentence-Plan(node bound to $v$)
                if $s =$ "fail"
                    then backtrack to (2)
                    else attach $s$ to $p$ at $v$
            return $p$
```

Figure 9.7: The sentence-planning algorithm. It outputs the most preferred complete well-formed SemSpec for a subgraph rooted at given node in the SitSpec.

Now, our solution to the theoretical issue is to assume that no complex interaction between decisions takes place. So, assuming that each option has an associated numeric score (related to the input preferences), we can simply choose the set of options that would maximize the sum of the scores, subject to the other constraints of building a proper SemSpec. We leave it to future work to formalize how the context affects the combination of the scores.

With respect to the practical issue, to avoid an exhaustive search through all of the complete well-formed sentence plans, we use the following heuristic, adopted from (Stede, 1999):

> In order to find the *globally* preferred sentence plan, make the most preferred *local* choices.

That is, whenever a (local) decision is made between several options, choose the option with the highest score. Thus, we postulate that the most preferred sentence plan will be one of the first few sentence plans generated, though we offer no proof beyond intuition at this point.

Our sentence-planning algorithm, shown in 9.7, implements the above ideas, but it is designed in a general fashion so that it can be used as a framework for testing future strategies and solutions (to the above problems). The algorithm is a modification of MOOSE's algorithm for finding and building the most preferred complete well-formed SemSpec. The main difference is the two-tiered selection process, indicated by (1) and (2) in the algorithm. (Note, however, that this version of the algorithm does not show how complete coverage or well-formedness is ensured.) The algorithm recursively builds a SemSpec, starting from a given SitSpec node, and returns either the SemSpec or "fail". It starts with the first (most preferred) cluster option associated with the node and the first (most preferred) candidate near-synonym of that cluster option. Given a candidate, the algorithm returns either the candidate's PSemSpec, if the PSemSpec has no external variables (which terminates the recursion), or a SemSpec built by attaching sub-SemSpecs, one for each external variable, to the PSemSpec. These sub-SemSpecs are built by recursively calling the algorithm on each node bound to an external variable. If any of the recursive calls fails, then the algorithm backtracks and chooses the next best near-synonym. If it runs out of near-synonyms in the cluster option, it backtracks even further and chooses the next best cluster option.

We implemented this algorithm in I-Saurus. As an example, figure 9.8 depicts the state of I-Saurus continuing on from figure 9.4. It shows that the cluster options for `say_C`, `John_C`, and `lie_C` were chosen, and that the candidates *tell*, *John*, and *lie* were chosen for each of the cluster options, respectively. The corresponding PSemSpecs are shown at the bottom, unified into a complete well-formed SemSpec.

Now, if we want to rank several sentence plans in terms of preference, then after the algorithm returns a plan, we can backtrack to (2), and continue building another plan. Once we have a set of sentence plans, we can compute for each one a score that indicates how well it satisfies all of the preferences. Then we can sort the sentence plans by this score. Following our assumption of no interaction, we define the score as follows:

$$Satisfaction(S,P) = \sum_{w \in S} Osat(P,w) \tag{9.1}$$

where S is a sentence plan consisting of a set of words, P is a set of preferences. *OSat* is defined below; it returns the contribution made by a word to satisfying the set of preferences. Thus, the most preferred sentence plan S' is the one whose set of word choices maximizes $Satisfaction(S',P)$. This function accounts for tradeoffs in the satisfaction of preferences, because it finds the set of words that collectively satisfy as many of the preferences as possible, each to the highest degree possible.

Now we must specify how the local decisions are made. We need algorithms for ‘Next-Best-Cluster-Option’ and ‘Next-Best-Near-Synonym’ of the sentence-planning algorithm. Formalizing the former process is outside the scope of this thesis, but we can assume that an algorithm will in due course be devised for ranking the cluster options according to criteria supplied in the input. In fact, MOOSE can rank options according to preferences to foreground or background participants (in order to make them more or less salient), but it is only a start. For now, I-Saurus uses an arbitrary ranking.

This thesis is concerned with the second type of local decision—choosing the most preferred near-synonym of a cluster—which finally brings us to the lexical-choice algorithm.

9.5 The lexical-choice algorithm

Once a cluster has been selected, each of its near-synonyms becomes a candidate for choice. This section describes our algorithm for choosing the most appropriate candidate.

9.5.1 Satisfying preferences locally

Informally, given a set of candidate near-synonyms and a set of preferences (both local and global), the idea is to find the near-synonym that best satisfies the most preferences. This problem is difficult for two reasons. First, preferences can be satisfied to different degrees by each of the near-synonyms in different ways. Second, it is not obvious how to combine, for a given word, the degrees to which it satisfies each of the preferences. And this assumes that the preferences are compatible with one another to begin with (see below).

To start, for each preference that can potentially be satisfied in the cluster, we would like to compute for each near-synonym a numeric *satisfaction score* that indicates how well it satisfies the preference, if at all. Depending on the type of preference, the following factors might be

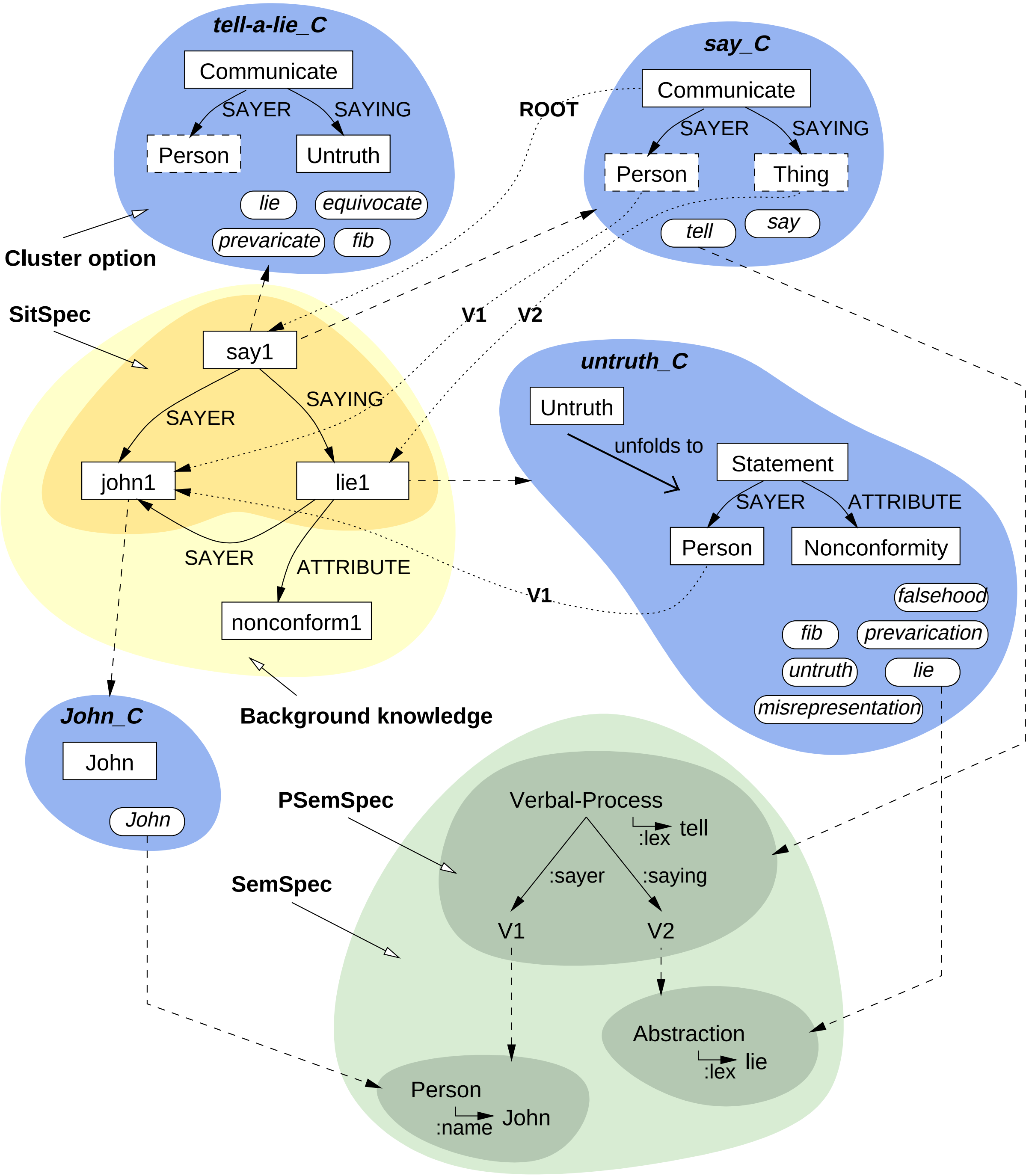


Figure 9.8: The state of I-Saurus after a SemSpec has been constructed. First, the `say_C` cluster option was chosen. From that, *tell* was chosen. Its PSemSpec required SemSpecs to be built for the nodes bound to the variables `V1` and `V2` (i.e., `john1` and `lie1`, respectively). SemSpecs for these nodes were built after choosing the words *John* and *lie*, and are shown. Attaching these two SemSpecs to the PSemSpec for *tell* results in the final SemSpec.

involved in computing a satisfaction score. (Recall that each lexical item in a cluster is represented by a set of distinctions, at most one of which will correspond to the preference.)

- Semantic preferences:
  - The frequency with which the near-synonym expresses the (preferred) concept.
  - The similarity between the preferred indirectness and actual indirectness of expression of the concept.
  - The degree to which the preferred concept and the concept of the distinction are similar.
- Expressive preferences:
  - The frequency with which the near-synonym expresses the (preferred) attitude towards the entity.
  - The similarity between the preferred attitude and the actual attitude that the near-synonym expresses towards the entity.
- Stylistic preferences:
  - The similarity between the preferred value and the actual value that the near-synonym has on the specified stylistic dimension.

Then, with the individual satisfaction scores for a particular candidate in hand, we can compute an overall satisfaction score for the candidate by combining its individual scores. Finally, we choose the candidate with the highest overall satisfaction score. Since each distinction situates a near-synonym on a particular dimension, this computation is analogous to finding the near-synonym that is closest to the set of preferences in a multi-dimensional space. Of course, it is not as simple as this, because preferences and distinctions have a structure that involves different continua (for frequency and indirectness) and different kinds of dimension, as well as structured concepts.

In the best case, in which one near-synonym (or even several) satisfies all of the preferences, it would be easy to choose the most appropriate candidate. But what happens when no single word simultaneously satisfies all of the input preferences? For instance, there is no near-synonym of *error* that satisfies both a preference to imply stupidity and one to imply misconception; *blunder* satisfies the former but not the latter, and *mistake* vice versa. Similarly, there is no informal word for an untrue statement (i.e., a *lie*) that also expresses that the lie is insignificant; *fib* is an option, but it is not a formal word. And, there is no word for a tract of trees that is both large and not wild; *forest* has the former property, *woods* the latter.

We can resolve such situations in two different ways. First, we can satisfy one of the preferences with an appropriate choice in the current cluster, and satisfy the other preferences (that have been sacrificed by this decision) with simultaneous choices in other clusters. Second, we can simply ignore one or more of the preferences. Our sentence-planning algorithm attempts to do the former, because it looks for the set of words that collectively satisfies as many of the preferences as possible. But when it isn't possible, the algorithm, in effect, ignores a preference. Of course, importance factors can be used to control which preferences receive higher priority.

Finally, we must be careful here, because sometimes the reason that we can't find a way to simultaneously satisfy two preferences might not be because there is no such word but because the preferences themselves are incompatible. That is, it might be impossible to satisfy the preferences under any circumstances. We will now briefly discuss this issue.

9.5.2 Incompatible preferences

There is an important issue with respect to representing lexical variation that we have put off discussing until now, because it becomes particularly relevant at this point. We have assumed up to now that the set of preferences on the input is consistent or well-formed. But two preferences (or two lexical distinctions of the same word) can be *incompatible*, and there are different ways in which they can be incompatible. For instance, a preference for low severity is incompatible with a preference for high severity. Not only is it impossible for a word to simultaneously express both ideas, but if the system were to attempt to satisfy both, it might output a dissonant expression[3] such as "I (gently) chided Bill for his (careless) blunder" (i.e., the preference to harshly criticize Bill is satisfied by *blunder*, and the preference to gently criticize Bill is satisfied by *chide*). Although, sometimes a dissonant expression might be desirable.

Now, often it is reasonable to assume that the input will be well-formed. For instance, in the context of MT, we can reasonably assume a 'good' analysis stage would output only well-formed expressions free of incompatibilities. However, we would still like our system to be able to detect incompatibilities for two reasons. First, we would simply like to be able to detect input that is not well-formed; second, we would like to be able to verify the coherency of lexical representations—detecting incompatibilities in preferences is analogous to detection incompatibilities in lexical distinctions.

So, we have identified the following types of incompatibility, only the first of which our system can currently handle:

Simple incompatibility Preferences to express different values on the same dimension are usually incompatible. For instance, preferences for both low and high severity, or preferences for both official and arbitrary authority, are incompatible. This kind of incompatibility is easy to detect in our formalism, because peripheral concepts are explicitly modelled as dimensions.

Denotational incompatibility A denotational incompatibility occurs when the denotations of two preferences are logically inconsistent. For instance, it is probably inconsistent to think of someone as blameworthy who causes an error out of a misconception (i.e., non-volitional acts are not blameworthy, however this may be cultural). In other words, 'misconception' logically entails 'non-blameworthiness'. Thus, preferences for implying both misconception and blameworthiness are incompatible. Ideally, an ontology would encode such relationships, and reasoning about them should therefore be possible in our model.

Contextual incompatibility A contextual, or pragmatic, incompatibility is like a denotational incompatibility, except that the context has a more obvious influence on reasoning about it. For instance, it is only sometimes inconsistent to think of someone as stupid who causes an error out of a misconception. The context will help to determine whether 'misconception' logically entails 'stupidity'.

Extensional incompatibility Can a tract of trees that is large, far from civilization, and yet not wild exist? From our knowledge of the world—about trees, forests, and so on—we know such a thing would be rare, if it could exist at all. So, preferences for large size, low urban proximity, and non-wildness are in a sense incompatible, but not because of their inherent

[3] Dissonance is one form of semantic anomaly that Cruse (1986) defines by example: such as "Arthur is a married bachelor."

semantics but because no entity (or extension) is known to exist that satisfies all of the preferences. One might discover, one day, such a tract of trees.

9.5.3 Formalization

Before we formalize the lexical-choice algorithm, there are two remaining issues to consider. First, if two or more near-synonyms satisfy the preferences to the same degree, then we need to decide how to choose one of them. What we need is a default preference. It could be to choose randomly, or it could be to choose according to a criterion such as frequency or typicality of usage, among others. In the system we assume that the candidate near-synonyms are ordered according to a pre-specified criterion, so, all else being the same, the candidate earlier in the list will be chosen.

Second, the process should be responsive to outside constraints. That is, a constraint such as a collocational constraint arising from another decision made by the system might automatically rule out some of the candidates. To account for such constraints, a cluster option maintains a list of candidate near-synonyms. If a constraint rules out a candidate, it is removed from the list. (The list is also useful when the system has to backtrack and make a new choice from the set of leftover candidates; the previous choice can simply be removed from the list.)

Now, formally, given

- a cluster C;
- a list W of n candidate near-synonyms of C, ordered according to a pre-specified criterion;
- an (infinite) set $\mathcal{P}$ of preferences, and a set $P \subset \mathcal{P}$ of m compatible preferences that can each be potentially satisfied by an appropriate choice in C, and which have varying importance: $Imp : P \rightarrow [0,1]$;
- a preference satisfaction function that returns a value that indicates the degree to which a word satisfies a preference: $Sat : \mathcal{P} \times W \rightarrow [0,1]$; and
- an overall satisfaction function that returns the degree to which a word satisfies a set of preferences: $Osat : 2^{\mathcal{P}} \times W \rightarrow [0,1]$

find the candidate $w \in W$ such that $Osat(P,w)$ is maximized. If two or more candidates should maximize the function, then the one that occurs earlier in W is returned.

We reduce this problem to an approximate-matching algorithm: for each candidate w, we determine the degree to which its set of distinctions matches the set of preferences P. Matching occurs in two steps. First, the algorithm finds for each preference $p \in P$, the lexical distinction of w that matches p, if one matches at all, and computes the degree of the match by computing $Sat(p,w)$. Second, assuming that the overall match score depends on the individual match scores, the algorithm computes $Osat(P,w)$ by combining the set of scores $Sat(p,w)$ for all $p \in P$. Various functions are available for combining the scores including simple functions like a weighted average or a distance metric, and more complex functions, which could, for instance, take into account dependencies between preferences.[4] Deciding on a function is a subject for future research that will empirically evaluate the efficacy of various possibilities.

[4] For instance, we might want to consider a preference only after another preference has been satisfied (or not), or only to resolve conflicts when several words satisfy another preference to the same degree.

For now, we define *Osat* as a weighted average of the individual scores, taking into account the importance factors:

$$Osat(P,w) = \sum_{p \in P} \frac{Imp(p)}{m} Sat(p,w) \tag{9.2}$$

The next section discusses how to compute $Sat(p,w)$.

9.5.4 Computing preference satisfaction scores

We would like to reduce the problem of calculating how well a word *w* satisfies a given preference *p* to one of computing similarity between lexical distinctions, since we have already defined a similarity function *Sim* that takes into account all of the factors that we discussed above (see chapter 8). That is, we would like to compute satisfaction as follows:

$$Sat(p,w) = Sim(d(p),d(w)) \tag{9.3}$$

where $d(p)$ is a distinction related to the preference, and $d(w)$ is the matching lexical distinction of *w*.

There are two steps in the reduction: (1) find the distinction $d(w)$ of the near-synonym *w* that matches the preference *p*, if there is one that matches (e.g., if the preference is to express a high degree of severity of criticism, we must compare that preference to the candidate's distinction on the peripheral concept of severity of criticism); and (2) convert the preference *p* to $d(p)$, a kind of *pseudo-distinction*, which has the same form as a regular lexical distinction, thus putting it on even ground for comparison to $d(w)$.

Finding the matching lexical distinction The procedure for the first step is different for each type of preference, but essentially we must compare each of the lexical distinctions of *w* to *p* in order to find the one that uses the same dimension (whether it is stylistic, attitudinal, or denotational).

Starting with the easy case, a stylistic distinction matches a stylistic preference if they refer to the same stylistic dimension. That is,

$$\begin{aligned} p &= (value_p \; dimension_p) \quad \text{and} \\ d(w) &= (w \; value_w \; dimension_w) \end{aligned}$$

match if $dimension_p = dimension_w$. So, we simply search through the stylistic distinctions of *w* until we find a match. A match will always be found, since every word has a default distinction (of "medium") on every stylistic dimension.

Expressive preferences are more complex. We must make sure that the entity about which the attitude is to be expressed is the one referred to by the variable in the expressive distinction. That is,

$$\begin{aligned} p &= (stance \; entity) \quad \text{and} \\ d(w) &= (w \; frequency \; strength \; attitude \; variable) \end{aligned}$$

match if *variable* is bound to *entity*. The procedure is to search through the expressive distinctions of *w* until we find one whose variable is bound to the entity. A match will always be found, since every word has a default "neutral" attitude. If the variable is part of the core denotation then it would have been bound during the first stage of processing that instantiates the cluster options, so it is an easy matter to check. On the other hand, if the variable is declared in the 'p-link' field of the cluster, it might not have been bound yet, because the 'p-link' variables are accessed only when necessary. If this is the case, the system temporarily binds the variable to the entity. It then performs variable substitution on the appropriate constraint, and queries Loom to see if the constraint is true of the situation (i.e., if the facts are in the background knowledge). If the query returns true, then the binding is made permanent. For example, the following distinction matches the following preference:

Distinction:	`(lie_l usually medium pejorative V1)`
Preference:	`(favour john1)`

because when `V1` = `john1`, it satisfies the constraint `(:and (Person V1) (SAYER ROOT V1))`, where `ROOT` = `error1` (refer back to the sample input in section 9.2, and to the `untruth_C` cluster on page 216 of appendix B). Note that the actual attitude taken (i.e., favourable or pejorative) is not taken into account by this matching process, but instead by the similarity function *Sim* (see equation 9.3).

Denotational preferences involve the most work. Since denotational distinctions represent 'values' on various dimensions, as do preferences, we have to find the distinction of *w* that involves the same dimension as the preference. For example, if the preference involves the dimension of 'severity of criticism' with whatever value, then we have to find the lexical distinction, if there is one, that also involves 'severity of criticism'. Peripheral concepts in the cluster define the dimensions of variation, so in a sense, they are the necessary applicability conditions for denotational distinctions. So,

$$p = (\textit{method sitspec}) \quad \text{and}$$
$$d(w) = (\textit{w frequency strength indirectness configuration})$$

match if the peripheral concept of the *configuration* matches *sitspec*, or actually, the concept of which *sitspec* is an instance. In other words, the latter must be a specialization of the former. We can use the same matching procedure that we use for matching core denotations to SitSpecs, since peripheral concepts and preferences are represented in the same formalisms. So, the procedure is to search through all the denotational distinctions and find the one whose peripheral concept matches. (Note that, as with expressive distinctions, we also have to check the 'p-link' constraints on the variables of the peripheral concept.) Since the default is that a word has an unknown distinction on any given peripheral concept, a matching distinction will only be found if one was explicitly entered into the lexicon. If not, $d(w)$ is given the value "unknown". For example, the following distinction matches the following preference:

Distinction:	`(fib_l always medium implication (P6 (DEGREE 'low)))`
Preference:	`(imply (significance (ATTRIBUTE-OF lie1) (DEGREE 'low)))`

Forming a distinction out of the preference In order to compare the preference p to $d(w)$, we form a pseudo-distinction $d(p)$ out of the preference as follows (assuming $d(w)$ matches p).

For stylistic preferences,

$$\text{if} \quad p = (\textit{value dimension})$$
$$\text{then} \quad d(p) = (\text{nil } \textit{value dimension}).$$

For expressive preferences, we set the frequency to “always” (since we assume that the preference is to convey the idea), the strength to “strong”, the attitude according to the preferred stance, and the variable to the variable that binds the entity. That is,

$$\begin{array}{rl} \text{if} & p = (\mathit{stance}\ \mathit{entity}) \\ \text{then} & d(p) = (\text{nil always strong } \mathit{attitude}\ \mathit{variable}), \end{array}$$

where

$$\mathit{attitude} = \begin{cases} \text{“favourable”} & \text{if } \mathit{stance} = \text{“favour”} \\ \text{“neutral”} & \text{if } \mathit{stance} = \text{“neutral”} \\ \text{“pejorative”} & \text{if } \mathit{stance} = \text{“disfavour”} \end{cases}$$

$$\mathit{variable} = \text{the variable that binds } \mathit{entity}.$$

For denotational preferences, we set the frequency and strength to “always” and “medium”. We set the indirectness according to the preferred method, and we set the configuration to the concept of which the *sitspec* is an instance. That is,

$$\begin{array}{rl} \text{if} & p = (\mathit{method}\ \mathit{sitspec}) \\ \text{then} & d(p) = (\text{nil always medium } \mathit{indirectness}\ \mathit{configuration}), \end{array}$$

where

$$\mathit{indirectness} = \begin{cases} \text{“suggestion”} & \text{if } \mathit{method} = \text{“suggest”} \\ \text{“implication”} & \text{if } \mathit{method} = \text{“imply”} \\ \text{“denotation”} & \text{if } \mathit{method} = \text{“denote”} \end{cases}$$

$$\mathit{configuration} = \text{ a configuration of concepts that } \mathit{sitspec} \text{ is an instance of.}$$

What is interesting here is building a configuration of concepts out of the *sitspec*, which represents an instance. We have to do this since lexical distinctions involve conceptual configurations and not instances of conceptual configurations. To do this, we traverse the specification and build a conceptual configuration with an ‘equivalent’ structure by replacing the nodes of the specification with their concepts, and by adding the appropriate variables (that we can determine from the bindings stored in the cluster option). For example, the following two specifications would be converted into the following two configurations, respectively:

```
(intend1                       (P2 Intend
  (ACTOR john1)          ⟹      (ACTOR (V1 John))
  (ACTEE mislead1))              (ACTEE (P2-1 Mislead)))


(criticism1                    (P4 Criticism
  (ACTEE john1)                  (ACTEE (V1 John))
  (ATTRIBUTE             ⟹      (ATTRIBUTE
     (severe1                       (P4-1 Severity
       (DEGREE 'high))))              (DEGREE 'high))))
```

	Candidate					
Preference	*fib*	*lie*	*misrepresentation*	*untruth*	*prevarication*	*falsehood*
1 Insignificance	1.00	0.00	0.00	0.00	0.00	0.00
2 Deliberateness	0.50	1.00	0.75	0.25	0.75	0.00
3 Disfavour	0.50	0.63	0.50	0.50	0.50	0.50
4 Low formality	1.00	0.50	0.50	0.50	0.00	0.50
Overall Score	3.00	2.13	1.75	1.25	1.25	1.00

Table 9.2: Similarity scores and overall satisfaction scores between four preferences and the six candidates of the *error* cluster.

9.5.5 Example

Returning to the situation involving John and his lie, consider the following four preferences to convey various ideas:

1. `(imply (significance1 (ATTRIBUTE-OF lie1) (DEGREE 'low)))`

2. `(imply (intend1 (ACTOR john1) (ACTEE mislead1)))`

3. `(disfavour john1)`

4. `(low formality)`

If during processing, the `untruth_C` cluster option were chosen, then the system would choose the noun *fib* (with *lie* ranked second), because its overall satisfaction score is highest. Table 9.2 tabulates the results for each candidate in the cluster.

Expanding on the scores for $w_1 = fib$ and $w_2 = lie$, the following distinctions were compared:

1. $d(p_1) =$ (nil always medium implication (P6 Insignificance . . . 'low))
   $d(w_1) =$ (fib_l always medium implication (P6 Insignificance . . . 'low))
   $d(w_2) =$ unknown

2. $d(p_2) =$ (nil always medium implication (P2 Intend . . .))
   $d(w_1) =$ (fib_l sometimes medium implication (P2 Intend . . .))
   $d(w_2) =$ (lie_l always medium implication (P2 Intend . . .))

3. $d(p_3) =$ (nil always medium pejorative V1)
   $d(w_1) =$ (fib_l always medium neutral V1)
   $d(w_2) =$ (lie_l usually medium pejorative V1)

4. $d(p_4) =$ (nil low formality)
   $d(w_1) =$ (fib_l low formality)
   $d(w_2) =$ (lie_l medium formality)

9.6 Implementation notes

I-Saurus is implemented in Common Lisp and CLOS (the Common Lisp Object System). We defined classes for every major component of the system, including situations, preferences, cluster options, clusters, and lexical items. We did not define classes for SitSpecs or SemSpecs (though it would be useful to do so) because we used most the of code already provided in the MOOSE system (which does not use CLOS) for creating and manipulating these objects. In this section, we briefly discuss the important slots and methods of situations, cluster options, and preferences. (Clusters and lexical items were covered in chapter 7.)

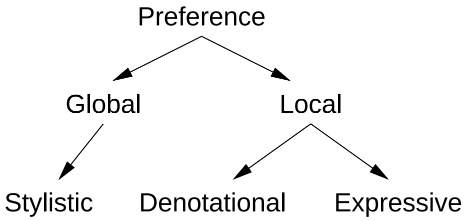
To run the system, a new situation object is instantiated and initialized with the input (SitSpec and preferences). The background knowledge (of Loom assertions) is assumed to be global knowledge in the system. Then, the situation's **Best-semspec** method is called to build a ranked set of sentence plans. Finally, the sentence plans are verbalized by Penman.

Situation The situation class is the main class of the system. Figure 9.9 shows the slots and methods of the class. Once instantiated, a situation initializes the system, instantiates and initializes all of required preferences and cluster options, and eventually runs the sentence-planning and lexical-choice algorithms. A new situation is instantiated for each new sentence to be generated.

Cluster option A cluster option is instantiated for each cluster that can potentially participate in the building of the output. Figure 9.10 shows the slots and methods of the class. The main task of a cluster option is to run a lexical chooser for its near-synonyms. In effect, each cluster option is running a separate chooser concurrently, and outputs the current best choice, subject to the preferences and constraints. In reality, the SemSpec building algorithm calls upon the cluster option to make a choice when the time is appropriate. Since the algorithm backtracks when it finds that a well-formed SemSpec cannot be built using the decisions made so far, it will ask the cluster option to make another choice (i.e., the next best choice).

Preference

From a generic preference class, we defined two subclasses (for global and local preferences), and then three more subclasses corresponding to the three types of preference, as shown here:



The purpose of a preference object is to maintain status about how well it has been satisfied by various choices made by the algorithm. In the current system, only lexical choices (made by cluster options) affect the status of preferences. Figure 9.11 shows the slots and methods of the class.

SITUATION:

Slots:

SitSpec The SitSpec of the situation.

Cluster-options A list of cluster options whose core denotations cover (or match) some part of the SitSpec.

Local-preferences A list of local preferences of the situation.

Global-preferences A list of global preferences of the situation.

Methods:

Initialize (situation, tree, prefs) Given a SitSpec **tree** and a set of preferences **prefs**, initializes the **situation** by calling **Traverse** and **Add-preferences**.

Traverse (situation, tree) Creates the list of cluster options by traversing the SitSpec **tree** starting at the root, and at each node finding the set of clusters whose core denotations match the subgraph rooted at that node. This is the same code used in MOOSE, with a few minor modifications to replace MOOSE's verbalization options with cluster options. Each matching cluster instantiates a cluster option (see the cluster option class).

Add-preferences (situation, prefs) Parses each input preference, and instantiates a preference object for each (see the preference class).

Best-semspec (situation, n) Using the algorithm outlined in figure 9.7, builds and returns the first **n** SemSpecs for the **situation**, sorted in decreasing order of overall satisfaction score.

Report (situation) Calculates and returns the overall satisfaction score of all the preferences, after a complete SemSpec has been built.

Figure 9.9: Slots and methods of the situation class.

CLUSTER OPTION:

Slots:

Cluster A pointer to the cluster that the cluster option ‘instantiates’.

Situation A pointer to the situation for which this is an option.

Covers A list of SitSpec nodes that the option covers. (Initialized during matching.)

Bindings A list that binds cluster variables to their associated SitSpec nodes. (Initially set up during matching; bindings are added during processing.)

Preferences A list of local preferences that can potentially be satisfied by a lexical choice from the cluster option.

Lex-options A list of current lexical options, i.e., candidates for choice.

Choice The current lexical choice of the cluster option.

Methods:

Initialize (c-option) Sets **covers** and **bindings** of the option (after it has been determined that the core denotation of the cluster matches part of the SitSpec).

Init-chooser (c-option) Put the chooser in its initial state by copying the list of near-synonyms from the cluster into the **lex-options** slot.

Next-Best-Near-Synonym (c-option) Using the algorithm discussed in section 9.5, finds the word in **lex-options** that best satisfies the local and global preferences, removes it from **lex-options**, and puts it in **choice**. Individual satisfaction scores for each preference are stored with the preference. If the system is backtracking and asking the cluster option to make a new choice, the method retracts the old choice first.

Figure 9.10: Slots and methods of the cluster option class.

PREFERENCE:

Slots:

Situation A pointer to the situation for which this is a preference.

Importance The importance, from 0–1, of the preference.

Satisfaction A list of cluster options from which choices have been made that have satisfied the preference in some way or other.

Cluster-options A list of cluster options that can satisfy this preference (for local preferences only).

Methods:

Initialize (preference, c-options) If **preference** is local, finds all the cluster options (which have already been found to match the SitSpec) that could potentially satisfy the preference. The preference is then added to the **preferences** list of each of these cluster options, and each cluster option is added to the preference's **cluster-options** list. For semantic preferences, the matching code (for matching core denotations to SitSpecs) is re-used. This initialization performs step one of 'computing preference satisfaction scores' (see section 9.5.4) ahead of time, so that it is only done once, increasing efficiency. As a side effect, we can determine if any preferences are incompatible with one another (e.g., if they ask for opposite values on the same dimension).

Satisfy (preference, c-option, lex-item, score) Adds the chosen **lex-item** with its satisfaction **score** to the **satisfaction** list.

Figure 9.11: Slots and methods of the preference class.

9.7 Summary

This chapter described our lexical-choice process, which takes advantage of the fine-grained lexical knowledge that we can now represent in the lexicon (using our clustered model of lexical knowledge). We formalized several criteria for lexical choice as preferences to express certain concepts in certain ways, to express attitudes, and to establish style. The lexical-choice process chooses the near-synonym (in a cluster) that best satisfies the most preferences, which are supplied as input to the process. The process uses an approximate-matching algorithm, which outputs a numeric value that represents how well the set of lexical distinctions of a particular word matches a set of preferences.

Even though many details of the lexical-choice and sentence-planning algorithms are preliminary, we were able to implement the algorithms in I-Saurus, a prototype sentence-planning system based on MOOSE. The next chapter evaluates I-Saurus, with the purpose of demonstrating that our clustered model of lexical knowledge adequately accounts for near-synonymy.

Chapter 10

Evaluation and testing

Now that we have developed a model for representing near-synonyms and differences between near-synonyms, we must evaluate it to demonstrate that it meets our objectives. That is, does a clustered model of lexical knowledge account for the phenomena involved in fine-grained lexical meaning? Which is to say, does it solve the problems of representing near-synonyms (see section 3.8)? While we can argue that it does solve these problems—as we have done—to properly substantiate this claim requires that we evaluate the model in the context of a system that uses the model. Therefore, we will evaluate I-Saurus to determine if it can choose the most appropriate words when given any combination of input preferences.

10.1 Evaluating NLP systems

Now, in evaluating our system we would like to draw on established methods of evaluating natural language processing (NLP) systems. But, in general, the evaluation of NLP systems has proven to be so difficult—indeed, only in the past decade, as systems have become increasingly powerful, have researchers begun to grasp the full complexity of the task (Lehrberger and Bourbeau, 1988; Hutchins and Somers, 1992; Arnold, Sadler and Humphreys, 1993; Sparck Jones and Galliers, 1996)—that comprehensive evaluation methodologies are just now being developed. For instance, in their survey of NLP evaluation technology, Sparck Jones and Galliers (1996) report that existing methodologies are task-specific (i.e., they are motivated by the particular application of the NLP system, be it information retrieval, message understanding, or whatever), and do not adequately define the subcomponents of the system (which are usually thought of as generic components—such as our lexical choice process). There has been no significant work done on evaluating generic components in isolation of a specific system. Sparck Jones and Galliers conclude that there is no 'magic bullet' for NLP evaluation—each case requires its own design—and recommend that for an evaluation to be worthwhile and repeatable, it must proceed by a meticulous identification of both the goals of the evaluation and the performance factors of the system.

Of course, the complexity of evaluating NLP systems arises out of the nature of language itself. The 'quality' of any particular text or linguistic expression is highly subjective, so judging the quality of the linguistic output of a system cannot usually be done automatically—human judges are required.[1] Consider how one might evaluate the quality of lexical choice in an MT

[1] Of course, there are many more criteria on which one could evaluate a system, including coverage of linguistic phenomena, speed, efficiency, extendibility, usability, and so on.

system. Generally, MT systems in development are evaluated in terms of quality and coverage (Arnold, Sadler and Humphreys, 1993; Nyberg, Mitamura and Carbonell, 1994). With respect to evaluating quality, the three main dimensions are *intelligibility*, *fidelity* (or accuracy), and *fluency* (or style). For lexical choice, we are most concerned with the latter two: does the lexical choice process choose the word that appropriately conveys the desired meaning and style? But since evaluating quality is subjective (especially for fluency), the usual design is to first assemble a very large and representative suite of test materials, which is a huge task in itself, and then to ask human judges to rate the quality of the output according to various measures. But while some objective measures have been developed for intelligibility and fidelity (such as error rate and reading time), none have yet been identified for fluency—partly because of the nature of fluency, and partly because there hasn't been much point in evaluating fluency when it's known to be poor for MT systems in general. So, the methods have not been developed to the point where they can be applied to a lexical choice process such as ours.

Objective (and quantitative) evaluation is, however, possible in some instances where we can precisely and objectively define the right and wrong behaviour. Such cases arise for systems that have a broad but shallow coverage of many linguistic phenomena (rather than a deep coverage of relatively few phenomena). For instance, we were able to automatically evaluate our statistical lexical-choice process using quantitative methods because the process was able to make a choice in every possible context, and because the nature of the task allowed for an objective evaluation criterion. But as we discussed in section 6.2.2, coming up with a 'gold standard' for typicality is an open problem.

In any case, the preceding discussion has assumed that the system under evaluation is in an advanced state of development, not a prototype system with a small experimental lexicon, like I-Saurus. It's a Catch-22: we can't properly evaluate I-Saurus until we have developed it into a real system with a large lexicon, but developing a real system would be so labour-intensive, in this case, that we wouldn't want to do it unless we knew beforehand that its model (of lexical knowledge) was adequate; and to know this, we would have to evaluate the system.

Therefore, although an evaluation methodology and a suite of test materials would be valuable contribution to this area of research, we feel that it is premature to work in this direction now. Not only is developing such a methodology a substantial undertaking in itself (involving new research), but there is no current system, including I-Saurus, in a position to be properly evaluated. We do believe, though, that a methodology should be developed in tandem with future research on fine-grained lexical representation and lexical choice. For now, we will rely on more informal methods.

10.2 Evaluating the clustered model of lexical knowledge

Before we get to the evaluation of the system, we will discuss how we evaluated the clustered model of lexical knowledge on its own. Our evaluation seeks to answer the following questions about the model:

- Does the model account for, or cover, a wide range of phenomena involved in near-synonymy? That is, does it solve the problems of representing near-synonyms listed in section 3.8?

- Does it adequately account for the phenomena?

- Can the model be used effectively in a system (i.e., in computational lexical choice)?

- Does the model scale up?

We address these questions in the following two sections.

10.2.1 Coverage of linguistic phenomena

In section 7.5 (page 122), table 7.2 summarized the types of lexical variation that the model is intended to account for. We showed that it is able to account for many of the major types of variation, including variation in style, expressed attitude, the indirect expression of peripheral concepts, and the fuzzy expression of concepts. Some other types were accounted for in only a preliminary way, and others not at all, but we believe that the model can be extended to cover these and many other types of variation, especially those to do with emphasis, collocation, and context (see the discussion of future work in chapter 11). We also showed that the model was able to account for these types of variation because it incorporated solutions, some preliminary, to almost all of the problems of representing near-synonyms. The only major problem left unsolved is how to handle the influence of context.

Of course, the real concern is whether the model adequately accounts for these types of variation and phenomena—adequate enough, that is, to enable sophisticated lexical choice and other lexical processes in a NLP system. We claim that the clustered model does adequately account for near-synonymy, but we can give only partial support to this claim through an evaluation of I-Saurus, which is just one particular application of the model (see section 10.3 below). The successful evaluation of I-Saurus also answers, positively, the third question above.

10.2.2 Extendibility

There are two relevant issues with respect to scaling up the model. First, can it be used to build a full-scale lexicon (i.e., one with 50,000 or more entries)? And second, more practically, how much effort is required to add new entries and clusters? For the former, we refer the reader to the literature on knowledge-based natural language processing (see sections 4.1.1 and 4.2.5). To the extent that any knowledge-based model will scale up, so will the clustered model. The complexity that we have added to the conventional knowledge-based model is contained in the representations of individual clusters and does not spread, so to speak, between clusters. In other words, if our model works with one cluster, it should work with 50,000 clusters, presuming that a knowledge-based model will work at that scale. And the success, so far, of the Mikrokosmos project (Mahesh and Nirenburg, 1995) demonstrates that large scale knowledge-based applications are viable.

For the latter, adding new lexical entries and clusters is a matter of lexicography and knowledge engineering. For each new cluster, the following must be developed and represented:

- A suitable set of near-synonyms.
- New concepts, if the required concepts are not already in the ontology.
- The core denotation and peripheral concepts of the cluster.
- The distinctions between the near-synonyms of the cluster.
- A lexical entry for each near-synonym (including PSemSpec and collocational constraints).

- A surface-level lexical entry for each word with morphosyntactic properties (i.e., a Penman lexical entry).

The difficult part is developing the core denotation, peripheral concepts, and distinctions, but we believe the task to be equal in difficulty to that faced by lexicographers today. Admittedly, developing synonym usage notes is hard lexicography, but it can be done. Our computational usage notes are just more rigorous but less detailed than the usual usage notes (since the model is less expressive than natural language is). In fact, we envision an automatic (or semi-automatic) process for acquiring the data, or a tool to help lexicographers build well-formed and suitable cluster representations (see chapter 11).

10.3 Evaluating the lexical-choice process

10.3.1 Testing strategy

The purpose of our testing was both to provide evidence that the clustered model of lexical knowledge adequately accounts for near-synonymy, and to demonstrate that the system, particularly the lexical choice component, can make the appropriate choices when given any set of preferences.

Because of our focus on lexical choice, we were not concerned about testing whether the system creates well-formed and complete sentence plans (though it does, because it is based on MOOSE), or whether it accounts for the interaction of syntactic and collocational preferences with lexical choice (which it does not, because we have yet to develop that part of the system). Thus, we did not explicitly test the sentence-planning algorithm. However, it was indirectly tested as a result of running other tests. Furthermore, we were not concerned about whether or not our particular ontology and cluster representations were perfect, merely that they were reasonable renderings of the usage notes in our source dictionaries (Gove, 1973; Room, 1981; Hayakawa, 1994).

Our testing strategy was relatively simple, involving two kinds of test. First, in order to demonstrate the general capabilities of the system in producing complete sentences and in managing several cluster options and preferences at once, we created two situations (each consisting of background knowledge and a SitSpec) that would each cause the system to instantiate a number of cluster options. We then ran the system several times on the same situation with different preferences, attempting to make it generate all possible combinations of choices. The first of these situations tests the `error_C` and `err_C` clusters; we reported the results of the test in the introduction (see section 1.4.3), so we won't repeat them here. The second situation is more complex, involving the `untruth_C` and `tell-a-lie_C` clusters, among others; the results of these tests are given in section 10.3.2 below.

And second, not having a clear precedent for evaluating our type of lexical choice process, and not having the means to develop a proper test suite or to even perform such an evaluation, we instead performed a kind of diagnostic evaluation. So, we designed a series of test cases that would isolate various aspects of the lexical-choice algorithm and of the lexical representations. To perform the evaluation, we built SitSpecs (for the input) that would generate only a single word or phrase, rather than complete sentences; that way, only a single cluster option would be instantiated during the process. Then by running various tests, each involving different preferences, on the different SitSpecs, we are able to diagnose whether or not the system and model perform as they should.

In sections 10.3.3 to 10.3.9 below, we give the results of the series of tests. Each case involves a particular situation and a set of preferences to be given as input to the system. We then show the the top four or five phrases (or sentences) that the system outputs. In each case, the preferences are of equal importance, unless otherwise stated.

The representations of all of the clusters referred to below are given in appendix B.

10.3.2 General test of the system

We defined the following background knowledge and SitSpec:

Background knowledge:

```
(John john1) (Alcoholic alcoholic1) (Communicate order1) (Perform perform1)
(SAYER order1 john1) (SAYEE order1 alcoholic1) (SAYING order1 perform1)
(Communicate tell1) (Statement lie1) (Nonconformity nonconform1)
(ACTOR perform1 alcoholic1) (ACTEE tell1)
(SAYER tell1 alcoholic1) (SAYING tell1 lie1)
(SAYER lie1 john1) (ATTRIBUTE lie1 nonconform1)
(Authority authority1) (Official official1) (Peremptory peremptory1)
(Significance significance1) (Misconception misconceive1)
(Contradict contradict2) (Categorical categorical2)
```

SitSpec:

```
(order1 (SAYER john1)
        (SAYEE alcoholic1)
        (SAYING (perform1 (ACTOR alcoholic1)
                          (ACTEE (tell1 (SAYER alcoholic1)
                                        (SAYING lie1))))))
```

I-Saurus can generate 960 different sentence plans from this single SitSpec, including plans that realize the sentences “John commands an alcoholic to lie” and “John orders a drunkard to tell a fib”. I-Saurus can be so prolific because of the many possible combinations of the near-synonyms of the six clusters involved: `John_C` (one near-synonym), `alcoholic_C` (ten near-synonyms), `order_C` (six near-synonyms), `say_C` (two near-synonyms), `untruth_C` (six near-synonyms), and `tell-a-lie_C` (four near-synonyms).

Table 10.1 shows the variety of output that is possible when different combinations of preferences are input to the system. (Numbered preferences are listed under the table.) So, for example, if we input preference 3, `(high formality)`, the system outputs “John enjoins an inebriate to prevaricate”. The output appears stilted in some cases because no other parameters, such as desired verb tense, were given to Penman, and because the system has no knowledge of collocational constraints. Of course, we could have defined many other preferences (and combinations of preferences), but we chose these particular ones in order to show some of the interesting interactions that occur between the cluster options during processing. Also, these particular cases are not meant to be representative of what a user might normally ask of the system. Below, we discuss each of the labelled cases.

(i) **Input prefs.:** 2 `(medium formality)`
4 `(high concreteness)`

Output: “John directs a drunkard to tell a lie.”

In this case, three of the clusters (`order_C`, `alcoholic_C`, and `untruth_C`) can make choices that satisfy both preferences, so in effect, the choices reinforce each other. That is, *direct*,

Case	Input preferences	Output
	None	John commands an alcoholic to lie.
	1	John commands a drunk to fib.
	2	John commands an alcoholic to lie.
	3	John enjoins an inebriate to prevaricate.
	4	John directs a drunkard to tell a lie.
	5	John commands a tippler to fib.
	6	John commands a drunk to lie.
	7	John commands an alcoholic to lie.
	8	John orders an alcoholic to lie.
	9	John commands an alcoholic to fib.
	10	John commands an alcoholic to tell an untruth.
	11	John commands an alcoholic to lie.
i	2, 4	John directs a drunkard to tell a lie.
ii	1, 9	John commands a drunk to fib.
iii	3, 6	John enjoins a drunkard to prevaricate.
iv	6, 10	John commands a drunkard to tell an untruth.
v	3, 9	John enjoins an inebriate to fib.
vi	3, 7, 9	John commands an inebriate to fib.
vii	3, 8, 9	John orders an inebriate to fib.
viii	3, 6, 8, 9	John orders a drunkard to fib.
ix	3, 5	John enjoins a tippler to tell a prevarication.
x	3, 5, 11	John enjoins a tippler to tell a prevarication.

Preferences:

```
1  (low formality)
2  (medium formality)
3  (high formality)
4  (high concreteness)
5  (favour alcoholic1)
6  (disfavour alcoholic1)
7  (imply (authority1 (ATTRIBUTE-OF john1) (ATTRIBUTE official1)))
8  (imply (authority1 (ATTRIBUTE-OF john1) (ATTRIBUTE peremptory1)))
9  (imply (significance1 (ATTRIBUTE-OF lie1) (DEGREE 'low)))
10 (imply (misconceive1 (ACTOR alcoholic1) (CAUSE-OF lie1)))
11 (imply (contradict2 (ACTOR lie1) (ATTRIBUTE categorical2)))
```

Table 10.1: Output of I-Saurus with different combinations of preferences.

drunkard and *lie* are all highly concrete words of medium formality. So, the system had no problem in making a choice.

(ii) **Input prefs.:**

```
1 (low formality)
9 (imply (significance1 (ATTRIBUTE-OF lie1) (DEGREE 'low)))
```

Output: "John commands a drunk to fib."

In this case, the choices don't reinforce each other, but they don't adversely affect each other either. *Fib* is informal and implies low significance (satisfying both preferences), *drunk* is informal, and *command* is one of the least formal words of its cluster. So the system is able to satisfy both preferences.

(iii) **Input prefs.:**

```
3 (high formality)
6 (disfavour alcoholic1)
```

Output: "John enjoins a drunkard to prevaricate."

This case is similar to (ii), but there is a slight clash in the choice of words. *Drunkard*, not a highly formal word, is nonetheless the most formal pejorative near-synonym of that cluster. *Prevaricate* satisfies both preferences because it is formal and slightly pejorative. The system could have satisfied the preferences in the opposite way by generating "John enjoins an inebriate to lie" (*inebriate* is formal, but not pejorative; *lie* vice versa), but the sentence it did generate has a slightly higher satisfaction score because both words contribute to the satisfaction of pejorativity.

(iv) **Input prefs.:**

```
6 (disfavour alcoholic1)
10 (imply (misconceive1 (ACTOR alcoholic1) (CAUSE-OF lie1)))
```

Output: "John commands a drunkard to tell an untruth."

Here, the system must choose carefully since one cluster can satisfy the first preference while another cluster can satisfy the second preference, but neither cluster can satisfy both preferences together. So, the system chooses *drunkard*, because it is pejorative, and *untruth*, because it implies a misconception. No other combination of choices from the two clusters could have simultaneously satisfied both preferences.

(v) **Input prefs.:**

```
3 (high formality)
9 (imply (significance1 (ATTRIBUTE-OF lie1) (DEGREE 'low)))
```

Output: "John enjoins an inebriate to fib."

This case illustrates a clash in the satisfaction of one of the preferences. *Fib* is chosen despite the fact that it is informal, because it is the only word that implies an insignificant lie. But the system compensates by choosing two other formal words: *enjoin* and *inebriate*.

(vi) **Input prefs.:**

```
3 (high formality)
7 (imply (authority1 (ATTRIBUTE-OF john1) (ATTRIBUTE official1)))
9 (imply (significance1 (ATTRIBUTE-OF lie1) (DEGREE 'low)))
```

Output: "John commands an inebriate to fib."

If we add a preference to case (v) to imply that John has official authority, then the system must choose *command* instead of *enjoin*, further sacrificing high formality.

(vii) **Input prefs.:**

```
3 (high formality)
8 (imply (authority1 (ATTRIBUTE-OF john1) (ATTRIBUTE peremptory1)))
9 (imply (significance1 (ATTRIBUTE-OF lie1) (DEGREE 'low)))
```

Output: "John orders an inebriate to fib."

Similar to (vi), if we add a preference to (v) to imply that John has peremptory authority, the system now chooses *order*, which is not a highly formal word either.

(viii) **Input prefs.:**

```
3 (high formality)
6 (disfavour alcoholic1)
8 (imply (authority1 (ATTRIBUTE-OF john1) (ATTRIBUTE peremptory1)))
9 (imply (significance1 (ATTRIBUTE-OF lie1) (DEGREE 'low)))
```

Output: "John orders a drunkard to fib."

Now, if we add a preference to case (vii) to disfavour the alcoholic, the system completely sacrifices high formality, because it chooses *drunkard*, which has medium formality. However, notice that it didn't choose *drunk*, which despite being more pejorative than *drunkard*, is informal. It is debatable whether the system should have sacrificed high formality like this—why not sacrifice another preference? In the end, formality could have been given more importance, if that were desired.

(ix) **Input prefs.:**

```
3 (high formality)
5 (favour alcoholic1)
```

Output: "John enjoins a tippler to tell a prevarication."

Compare this case to the case where just preference 3 is specified. It chooses *tippler* instead of *inebriate*, because *tippler* is favourable. It also prefers the noun *prevarication* to the verb *prevaricate*, because the latter is slightly pejorative (according to our particular representation of the words.)

(x) **Input prefs.:**

```
3 (high formality)
5 (favour alcoholic1)
11 (imply (contradict2 (ACTOR lie1) (ATTRIBUTE categorical2)))
```

Output: "John enjoins a tippler to tell a prevarication."

Now, when we add a third preference to case (ix), the system outputs the same sentence without satisfying the new preference at all. *Lie* (the verb or the noun) is the only word that would satisfy it, but it is neither formal nor favourable. So, the system made the best choice it could under the circumstances. Note, however, that "John enjoins a tippler to lie" was its second choice.

This set of test cases should demonstrate that the system can manage several cluster options at once and satisfy several preferences (to varying degrees) by making appropriate choices from the cluster options. As more preferences are specified, it sometimes makes questionable choices when the preferences clash in some way; however, that is to be expected (even from a human author). The user can vary the importance factors to have finer control over the output.

10.3.3 Choices that establish style

We now turn to the tests directed at isolated clusters.

The cluster of near-synonyms including *alcoholic, drunkard, boozer, drunk, inebriate, dipsomaniac, sot, lush, soak,* and *tippler* involves the stylistic dimensions of formality, floridity, and concreteness (as well as expressed attitude), so it is ideal for our purposes (see page 212 in appendix B). Consider the following very simple situation involving a single instance of the concept `Alcoholic`.

Background knowledge:

```
(Alcoholic alcoholic1)
```

SitSpec:

```
(alcoholic1)
```

Test case 1 Satisfy a stylistic preference when none of the near-synonyms in the cluster explicitly has a value on the stylistic dimension.

With a preference for low familiarity, all of the words should be ranked equally, because none of them explicitly has a value on this dimension. The system should rely on its default neutral value for each word. And, as the following output shows, all the words are ranked the same:

Input preferences: **Ranked output:**

```
(low familiarity)                              0.50   an alcoholic
                                               0.50   a drunkard
                                               0.50   a boozer
                                               0.50   a drunk
                                               0.50   an inebriate
```

Test case 2 Satisfy a single stylistic preference.

With a preference for low formality, the following is output, which is correct since the top four words do have low formality (in the lexicon):

Input preferences: **Ranked output:**

```
(low formality)                                1.00   a drunk
                                               1.00   a boozer
                                               1.00   a sot
                                               1.00   a lush
                                               0.50   a alcoholic
```

And with a preference for high floridity, the following is output, which is also correct:

Input preferences: **Ranked output:**

```
(high floridity)                               1.00   an inebriate
                                               1.00   a dipsomaniac
                                               0.50   an alcoholic
                                               0.50   a drunkard
                                               0.50   a boozer
```

Test case 3 Satisfy several stylistic preferences at once.

When both low formality and high floridity are asked for, the system must compromise since none of the highly florid words are informal.

Input preferences:

```
(low formality)
(high floridity)
```

Ranked output:

```
1.50   a boozer
1.50   a sot
1.50   a lush
1.00   an alcoholic
1.00   a drunkard
```

So, for instance, choosing *boozer* satisfies the former stylistic setting at a level of 1.0, and the latter at a level of 0.5. (There is no highly florid word with medium formality, which would have the same satisfaction score.)

When we add in a request for high concreteness, *boozer* remains the top choice, but *drunkard* moves up because it is more concrete than the other words.

Input preferences:

```
(low formality)
(high floridity)
(high concreteness)
```

Ranked output:

```
2.50   a boozer
2.00   a drunkard
2.00   a sot
2.00   a lush
1.50   a drunk
```

10.3.4 Choices that express attitude

Since the *alcoholic* cluster also has words that can express good or bad attitudes towards the alcoholic (the referent), we can continue with the same background knowledge and SitSpec as above.

Test case 4 Satisfy a preference to express an attitude (whether pejorative, neutral, or favourable) towards an entity who is the referent of the near-synonym.

The system performs as follows, when asked to express a pejorative attitude towards the alcoholic:

Input preferences:

```
(disfavour alcoholic1)
```

Ranked output:

```
1.00   a drunk
0.83   a drunkard
0.67   a lush
0.62   a sot
0.50   an alcoholic
```

Note that *sot* appears lower than *lush*. Although *sot* expresses a stronger pejorative attitude than *lush*, it is represented as doing so “usually” rather than “always” (making it a less satisfying choice).

When asked to express a favourable attitude, the system outputs the following:

Input preferences:

```
(favour alcoholic1)
```

Ranked output:

```
0.83   a tippler
0.62   a boozer
0.50   an alcoholic
0.50   an inebriate
0.50   a soak
```

To test the next case, we turn to a new situation involving the cluster of near-synonyms of *thin* (e.g., *slim*, *skinny*, and *slender*; see page 216 in appendix B). An attitude expressed by an adjective is usually expressed not towards the referent of the adjective (which is a quality), but towards the entity that the adjective modifies. And being a quality, an adjective's core denotation does not mention the entity. Thus, the system must check the 'p-link' constraints, which encode the relation of the entity to the quality. So, consider the following situation.[2]

Background knowledge:

```
(Thin thin1) (Person person1) (ATTRIBUTE person1 thin1)
```

SitSpec:

```
(thin1)
```

Test case 5 Try to satisfy a preference to express an attitude towards an entity that in fact is in some relation to the referent of the near-synonym.

Now, when we ask for a favourable attitude towards the person, the system outputs the following:

```
Input preferences:                         Ranked output:
(favour person1)                           0.75   a slender person
                                           0.63   a slim person
                                           0.38   a thin person
                                           0.13   a skinny person
```

Asking for a pejorative attitude give the expected result:

```
Input preferences:                         Ranked output:
(disfavour person1)                        0.63   a skinny person
                                           0.38   a thin person
                                           0.13   a slim person
                                           0.00   a slender person
```

And, asking for a neutral attitude moves the neutral word, *thin*, to the top of the list, pushing the others down:

```
Input preferences:                         Ranked output:
(neutral person1)                          0.75   a thin person
                                           0.50   a skinny person
                                           0.50   a slim person
                                           0.38   a slender person
```

[2] Note that this input should produce a sentence plan to express just the quality; however the Penman generator cannot generate a sentence (or fragment thereof) that does not also specify the entity to which the quality is ascribed. So, for demonstration purposes we specified in the lexical entries for each near-synonym a PSemSpec that includes a 'dummy' person that the quality is ascribed to, like so:

```
(x1 / Person :property-ascription (x / Sense-And-Measure-Quality :lex thin))
```

The other option would have been to specify a SitSpec that included the person (e.g., `(person1 (ATTRIBUTE thin1))`); however, this would not have given the proper output in the current system, because the system does not yet know the compositional rules for nouns and adjectives.

For the final test case on attitudes, we must introduce a new situation that invokes a cluster in which no attitudinal distinctions are represented. The cluster of *accompany* near-synonyms is such as cluster (see page 211 in appendix B). Consider the following situation that involves John accompanying Mary, and a number of peripheral ideas (which are relevant to the set of test cases in the next section).

Background knowledge:

```
(John john1) (Mary mary1) (Go-With accompany1)
(ACTOR accompany1 john1) (CO-ACTOR accompany1 mary1)
(Companionship companion1) (Courtesy courtesy1)
(Protect protect1) (Supervise supervise1)
```

SitSpec:

```
(accompany1 (ACTOR john1) (CO-ACTOR mary1))
```

Test case 6 Satisfy a preference to express an attitude towards an entity that none of the near-synonyms can.

When we ask the system to express a favourable attitude towards John, the system finds that none of the words expresses any sort of attitude towards John, so the satisfaction is zero for all of them:

Input preferences:

```
(favour john1)
```

Ranked output:

```
0.00   John accompanies Mary.
0.00   John attends Mary.
0.00   John conducts Mary.
0.00   John escorts Mary.
```

10.3.5 Choices on a binary denotational dimension

In this section, we test whether the system correctly handles binary denotational dimensions. As a side effect, the cases will also show why it is necessary to represent the dimensions by full-fledged concepts, rather than by simple features.

The accompany_C cluster is a good cluster to test these cases, because its near-synonyms have a variety of distinctions on binary dimensions. So these tests refer to the 'John accompanying Mary' situation, introduced above.

Test case 7 Satisfy a single preference on a binary dimension.

We must consider a few sub-cases involving a single preference. It can be the case that only one near-synonym conveys the preferred meaning, or that several near-synonyms do. In the latter case, the near-synonyms might convey the meaning equally, or one might be better at it than the others.

So, if we ask the system to imply that John and Mary are companions (or that there is companionship between them), the system chooses *accompany*, because that is the only near-synonym that satisfies the preference, as follows:

Input preferences:

```
(imply (companion1 (RELATUM1 john1)
                   (RELATUM2 mary1)))
```

Ranked output:

```
1.00   John accompanies Mary.
0.00   John attends Mary.
0.00   John conducts Mary.
0.00   John escorts Mary.
```

Similarly, the system chooses *chaperon* to satisfy a preference to imply that John supervises Mary, as follows.

```
Input preferences:                           Ranked output:
(imply (supervise1 (ACTOR john1)             0.31   John chaperons Mary.
                   (ACTEE mary1)))           0.00   John accompanies Mary.
                                             0.00   John attends Mary.
                                             0.00   John conducts Mary.
```

When two words both satisfy a preference equally, the system should rank them the same. The following test shows this result:

```
Input preferences:                           Ranked output:
(imply (protect1 (ACTOR john1)               1.00   John escorts Mary.
                 (ACTEE mary1)))             1.00   John convoys Mary.[3]
                                             0.00   John accompanies Mary.
                                             0.00   John attends Mary.
```

So, either *escort* or *convoy* could be chosen to imply protection.

And, in the following case, when we ask the system to suggest that the accompaniment is a courtesy, it ranks *escort* highest, followed by *attend*, even though both equally suggest courtesy:

```
Input preferences:                           Ranked output:
(suggest (courtesy1                          0.75   John escorts Mary.
          (ATTRIBUTE-OF accompany1)))        0.50   John attends Mary.
                                             0.00   John accompanies Mary.
                                             0.00   John conducts Mary.
```

The difference is because the former word "usually" and latter word only "sometimes" suggests courtesy.

Test case 8 Satisfy several preferences at once.

To test if the system can handle several preferences, we gave the system the following three preferences to satisfy:

```
Input preferences:                           Ranked output:
(imply (companion1 (RELATUM1 john1)          1.47   John escorts Mary.
                   (RELATUM2 mary1)))        1.00   John accompanies Mary.
(imply (protect1 (ACTOR john1)               1.00   John convoys Mary.
                 (ACTEE mary1)))             0.31   John attends Mary.
(imply (courtesy1
        (ATTRIBUTE-OF accompany1)))
```

The system chooses *escort*, because, even though it doesn't imply the companionship, it does sufficiently satisfy the other two preferences. Thus, the system had to compromise. Had we varied the importance factors of the preferences, a different word would have been chosen, as the following output shows:

[3] *Convoy* is a slightly awkward choice in this context. The system makes this choice because it doesn't have any knowledge about collocational constraints or selectional restrictions; however, if it did it would rank *convoy* lower.

Input preferences:

```
(0.5 imply (companion1 (RELATUM1 john1)
                       (RELATUM2 mary1)))
(0.25 imply (protect1 (ACTOR john1)
                      (ACTEE mary1)))
(0.25 imply (courtesy1
             (ATTRIBUTE-OF accompany1)))
```

Ranked output:

```
0.50   John accompanies Mary.
0.37   John escorts Mary.
0.25   John convoys Mary.
0.08   John attends Mary.
```

With companionship twice as important as the others preferences, *accompany* moves to the top.

Test case 9 Satisfy a preference whose specification (of meaning) is more specific than any near-synonym in the cluster can express.

If we ask the system to imply that John is carefully supervising Mary (as opposed to *merely* supervising her), then the system still finds that *chaperon* is the best word, though the score is lower than when we ask for mere supervision:

Input preferences:

```
(imply (supervise1 (ACTOR john1)
                   (ACTEE mary1)
                   (ATTRIBUTE carefully1)))
```

Ranked output:

```
0.26   John chaperons Mary.
0.00   John accompanies Mary.
0.00   John attends Mary.
0.00   John conducts Mary.
```

This is the expected behaviour, because the actual distinction is very similar to the preference: the distinction is underspecified with respect to carefulness, or any other attribute for that matter.

Test case 10 Try to satisfy a preference that does not match any of the cluster's peripheral concepts.

If we reverse the participants of the preference for supervision and ask the system to imply that Mary is supervising John, the system should not be able to find a match for the preference, so it should rank all of the options the same. It outputs the following:

Input preferences:

```
(imply (supervise1 (ACTOR mary1)
                   (ACTEE john1)))
```

Ranked output:

```
0.00   John accompanies Mary.
0.00   John attends Mary.
0.00   John conducts Mary.
0.00   John escorts Mary.
```

10.3.6 Choices on a continuous denotational dimension

For continuous denotational dimensions, we must test, in addition to the cases above, some additional cases.

Consider the following situation, which invokes the `error_C` cluster (see page 213 in appendix B, or refer back to figure 7.5 on page 105):

Background knowledge:

```
(John john1) (Deviation deviation1) (Activity error1)
(ACTOR error1 john1) (ATTRIBUTE error1 deviation1)
(Criticism criticism1) (Severity severe1) (ATTRIBUTE criticism1 severe1)
```

SitSpec:

```
(error1)
```

One of the peripheral concepts of the `error_C` cluster is the continuous dimension of 'severity of criticism', on which we will ask for various degrees to be expressed.

Test case 11 Satisfy a preference that is 'underspecified' with respect to the cluster (i.e., one that specifies a dimension but no value).

When we do not give a value on the dimension, the system behaves as expected. It ranks *error, mistake,* and *blunder* equally, because they each involve the dimension in some way, while the other *error* synonyms do not.

```
Input preferences:                           Ranked output:
(imply (criticism1 (ACTEE john1)             1.00   an error
        (ATTRIBUTE severe1)))                1.00   a mistake
                                             1.00   a blunder
                                             0.00   a slip
```

But perhaps the scores should not be 1.0, since the words don't exactly match the preference. This reflects a possible problem in the function for computing conceptual similarity. At present, the similarity function (see section 8.3) is adequate for relative comparisons, but for an absolute comparison we would want to say that *error* does not match perfectly the given preference (i.e., it should have a score less than 1.0) since *error* specifies a level of severity, whereas the preference leaves it unspecified. Or perhaps a default level of severity, say medium severity, should be assumed, which would make *error* a perfect match, but not its near-synonyms. Clearly, a more sophisticated similarity function is required.

Test case 12 Satisfy a preference for a numeric degree on the dimension.

If we specify a numeric value (0.6), the system is able to choose appropriately. The value of 0.6 is near the middle of the dimension of severity, so it ranks *error* first, as it implies a medium severity:

```
Input preferences:                           Ranked output:
(imply (criticism1 (ACTEE john1)             0.98   an error
        (ATTRIBUTE (severe1                  0.93   a blunder
                    (DEGREE '0.6)))))        0.87   a mistake
                                             0.00   a slip
```

Test case 13 Satisfy a preference for a fuzzy degree on the dimension.

Given the fuzzy concepts of `low`, `high`, and `(OR low medium)` the system ranks the near-synonyms appropriately, as the next three cases show:

```
Input preferences:                           Ranked output:
(imply (criticism1 (ACTEE john1)             1.00   a mistake
        (ATTRIBUTE (severe1                  0.95   an error
                    (DEGREE 'low)))))        0.90   a blunder
                                             0.00   a slip
```

Input preferences:

```
(imply (criticism1 (ACTEE john1)
        (ATTRIBUTE (severe1
                    (DEGREE 'high)))))
```

Ranked output:

```
1.00   a blunder
0.95   an error
0.90   a mistake
0.00   a slip
```

Input preferences:

```
(imply (criticism1 (ACTEE john1)
        (ATTRIBUTE (severe1
         (DEGREE '(OR low medium))))))
```

Ranked output:

```
1.00   an error
1.00   a mistake
0.95   a blunder
0.00   a slip
```

10.3.7 Choices on a multi-valued denotational dimension

On a multi-valued dimension, the test cases are similar to those above, but we must add a few more. We will use the order_C cluster, which has a multi-value dimension for authority; it can take the values Official or Peremptory (or Arbitrary, a subconcept of Peremptory in our ontology) (see page 215 in appendix B). Consider the following situation:

Background knowledge:

```
(John john1) (Mary mary1) (Communicate order1) (Perform perform1)
(SAYER order1 john1) (SAYEE order1 mary1) (SAYING order1 perform1)
(ACTOR perform1 mary1)
(Go-concept go1) (ACTEE perform1 go1) (ACTOR go1 mary1)
(Authority authority1) (Official official1) (Peremptory peremptory1)
(Arbitrary arbitrary1) (Quality q1)
```

SitSpec:

```
(order1 (SAYER john1)
        (SAYEE mary1)
        (SAYING (perform1 (ACTOR mary1)
                          (ACTEE (go1 (ACTOR mary1))))))
```

Test case 14 Satisfy a single preference.

The following two examples ask for official authority and peremptory authority to be implied. While all of the *order* near-synonyms shown imply some kind of authority, only *command* implies official authority, so it was correctly chosen in the first example, and only *order* implies peremptory authority, so it was also correctly chosen, in the second example.

Input preferences:

```
(imply (authority1 (ATTRIBUTE-OF john1)
        (ATTRIBUTE official1)))
```

Ranked output:

```
1.00   John commands Mary to go.
0.75   John orders Mary to go.
0.75   John enjoins Mary to go.
0.35   John bids Mary to go.
```

Input preferences:

```
(imply (authority1 (ATTRIBUTE-OF john1)
        (ATTRIBUTE peremptory1)))
```

Ranked output:

```
1.00   John orders Mary to go.
0.75   John commands Mary to go.
0.75   John enjoins Mary to go.
0.47   John bids Mary to go.
```

Test case 15 Satisfy a preference for a value on a dimension that is more specific than any value specified in a denotational distinction of the cluster.

To test this case, we ask for an implication of arbitrary authority (which is a subconcept of peremptory authority). In this case, there is no word in the cluster that implies arbitrary authority—only more general kinds of authority—so the system should rank highest the word that has the closest implication to arbitrary authority. In fact, the system treats this preference the same as it would treat a preference for peremptory authority, because the similarity function considers fillers of attributes to be exactly similar if one is a subconcept of the other (see chapter 8). So its ranks *order* highest as a perfect match. This shows a shortcoming in the similarity function.

Input preferences:

```
(imply (authority1 (ATTRIBUTE-OF john1)
       (ATTRIBUTE arbitrary1)))
```

Ranked output:

```
1.00   John orders Mary to go.
0.75   John commands Mary to go.
0.75   John enjoins Mary to go.
0.47   John bids Mary to go.
```

Test case 16 Satisfy a preference for a value on a dimension that is more general.

If we instead ask the system to imply a very general kind of authority (whose attribute is simply a quality), the system appropriately ranks all of the candidates the same (except for *bid* which only suggests authority).

Input preferences:

```
(imply (authority1 (ATTRIBUTE-OF john1)
       (ATTRIBUTE q1)))
```

Ranked output:

```
0.75   John commands Mary to go.
0.75   John orders Mary to go.
0.75   John enjoins Mary to go.
0.35   John bids Mary to go.
```

10.3.8 Choices involving fuzzy denotations

Here, we test how the system handles near-synonyms that have fuzzy denotations. We'll use the cross-linguistic cluster of *forest* near-synonyms (*forest*, *woods*, *Wald*, and *Gehölz*), which differ in a complex mixture of size, wildness, and urban proximity (see page 214 in appendix B). In the following tests, we'll demonstrate how different preferences on each of these dimensions select an appropriate word. A situation involving a tract of trees is as follows:

Background knowledge:

```
(Tree tree1) (Tract-Of-Land forest1) (CONTAINS forest1 tree1)
(Size size1) (Proximity prox1) (Wildness wild1)
```

SitSpec:

```
(forest1)
```

Test case 17 Satisfy single and several preferences involving fuzzy denotations.

First, if we ask the system to express a large tract of trees, the system ranks *forest* and *Wald* equally well:

Input preferences:

```
(denote (size1 (ATTRIBUTE-OF forest1)
        (DEGREE 'high)))
```

Ranked output:

```
1.00   a forest
1.00   a Wald (Ger.)[4]
0.90   a woods
0.80   a Gehölz (Ger.)
```

A medium tract of trees generates either *woods* or *Wald*:

Input preferences:

```
(denote (size1 (ATTRIBUTE-OF forest1)
        (DEGREE 'medium)))
```

Ranked output:

```
1.00   a woods
1.00   a Wald (Ger.)
0.90   a forest
0.90   a Gehölz (Ger.)
```

And a small tract of trees generates either *woods* or *Gehölz*:

Input preferences:

```
(denote (size1 (ATTRIBUTE-OF forest1)
        (DEGREE 'low)))
```

Ranked output:

```
1.00   a woods
1.00   a Gehölz (Ger.)
0.90   a Wald (Ger.)
0.80   a forest
```

In each of these cases, the system chose the appropriate words.

If we specify that the tract of trees is large, wild, and far from civilization, the system ranks *forest* and *Wald* highest, as one would we expect:

Input preferences:

```
(denote (size1 (ATTRIBUTE-OF forest1)
        (DEGREE 'high)))
(denote (wild1 (ATTRIBUTE-OF forest1)
        (DEGREE 'high)))
(denote (prox1 (ATTRIBUTE-OF forest1)
        (DEGREE 'low)))
```

Ranked output:

```
3.00   a forest
3.00   a Wald (Ger.)
2.60   a woods
2.40   a Gehölz (Ger.)
```

If we now specify that the tract of trees should be expressed as large, wild, but a medium distance from civilization, the system ranks *Wald* slightly above *forest*, which seems reasonable. (Considering only the English words, *forest* beats out *woods*, which is also reasonable.)

Input preferences:

```
(denote (size1 (ATTRIBUTE-OF forest1)
        (DEGREE 'high)))
(denote (wild1 (ATTRIBUTE-OF forest1)
        (DEGREE 'high)))
(denote (prox1 (ATTRIBUTE-OF forest1)
        (DEGREE 'medium)))
```

Ranked output:

```
3.00   a Wald (Ger.)
2.90   a forest
2.70   a woods
2.50   a Gehölz (Ger.)
```

Finally, a small tract of trees that is far from civilization is either a *woods* or a *Wald*, according to the system, but the scores are all close:

[4] Penman is an English sentence generator, so it does not know how to build real German expressions. So, for demonstration purposes, we had to treat the German words *Wald* and *Gehölz* as though they were English words.

Input preferences:

```
(denote (size1 (ATTRIBUTE-OF forest1)
        (DEGREE 'low)))
(denote (prox1 (ATTRIBUTE-OF forest1)
        (DEGREE 'low)))
```

Ranked output:

```
1.90   a woods
1.90   a Wald (Ger.)
1.80   a forest
1.80   a Gehölz (Ger.)
```

In this case, the preferences are perhaps incompatible, because there is probably no type of woodland (at least none that has a name) that meets the description. We consider incompatible preferences in the next section.

10.3.9 Choices involving incompatible preferences

In these tests, we want to see what the system does when given preferences that are incompatible in different ways. We will refer to situations that have already been described in the previous sections.

Test case 18 Satisfy two preferences for different values on the same dimension.

Consider the situation involving the error. If given preferences for both low and high severity of criticism, then the system reports that they are incompatible:

Input preferences:

```
(imply (criticism1 (ACTEE john1)
       (ATTRIBUTE (severe1
                  (DEGREE 'low)))))
(imply (criticism1 (ACTEE john1)
       (ATTRIBUTE (severe1
                  (DEGREE 'high)))))
```

Ranked output:

```
Incompatible preferences:
(CRITICISM1 ...  'LOW)
(CRITICISM1 ...  'HIGH)
```

It can do this, because it recognizes that the preferences refer to the same peripheral concept, and by definition a peripheral concept can be given only one value per word in a cluster.

Test case 19 Satisfy two preferences that are denotationally incompatible.

Again, consider the error situation. As we said in section 9.5.2, a preference for implying misconception and a preference for implying blameworthiness are probably denotationally incompatible. So, if they are both given as input, the system should give a warning like the one above. However, our system outputs the following:

Input preferences:

```
(imply (misconceive1 (ACTOR john1)
                     (CAUSE-OF deviation1)))
(imply (blameworthy1 (ATTRIBUTE-OF john1)))
```

Ranked output:

```
1.50   A mistake
0.50   A blunder
0.00   An error
0.00   A slip
```

Clearly, the system doesn't know about the relation between blameworthiness and misconception.

Test case 20 Satisfy two preferences that are extensionally incompatible.

With the cluster of *forest* words, there probably does not exist a kind of tract of trees that is large, far from civilization, and yet not wild. Yet the system, not realizing this, ranks the words according to how well they satisfy all of the preferences:

Input preferences:

```
(denote (size1 (ATTRIBUTE-OF forest1)
         (DEGREE 'high)))
(denote (wild1 (ATTRIBUTE-OF forest1)
         (DEGREE 'low)))
(denote (prox1 (ATTRIBUTE-OF forest1)
         (DEGREE 'low)))
```

Ranked output:

```
2.90   A Wald (Ger.)
2.80   A forest
2.80   A woods
2.60   A Gehölz (Ger.)
```

10.4 Discussion

The results of our diagnostic tests support our claim that the clustered model of lexical knowledge adequately accounts for many types of variation in near-synonyms. The model enables a lexical choice process to choose the most appropriate near-synonym given a set of preferences.

On the positive side, the results demonstrate, on the one hand, that it is possible to model fine-grained lexical distinctions using full-fledged concepts (rather than features), fuzzy concepts, and non–necessary-and-sufficient features, and, on the other hand, that such a model can make a text generation system (and by extension, an MT system), more expressive.

However, the tests show that there are a several areas where the model could be improved or further developed. The problems arise when several preferences are specified. If the preferences cannot all be satisfied, then the system must compromise, but since the system is incapable of reasoning about the relationships between lexical distinctions, its choices become arbitrary. While there is nothing intrinsically wrong with an arbitrary choice when a compromise is necessary, the system would benefit if it had some way of deciding how to compromise. In other words, we need to improve the sentence-planning component of the system. The system also requires a a better similarity function. While the current function works in many cases, it is too simplistic in comparing concepts that involve complex structures. Finally, the system cannot detect when preferences are potentially incompatible, so it might inadvertently generate semantically dissonant expressions. (However, as we discussed in section 9.5.2, incompatible preferences should not arise in practice if the input is well-formed—as it would be if it came from a 'good' analysis stage of an MT system.)

Furthermore, as we already said, the model does not take into account all forms of lexical variation, nor the influence of context. Our use of data about frequency (of expression) when generating text is a clumsy simplification in this respect.

Chapter 11

Conclusion

> The right word may be effective, but no word was ever as effective as a rightly timed pause.
>
> Mark Twain.

11.1 Computational lexical semantics and near-synonymy

Computational lexical semantics has grown into a major subfield of computational linguistics. After all, every natural language processing system needs some sort of lexicon, and for some systems, the lexicon is the most important component. Yet, real NLP systems today rely on a relatively shallow, but broad, coverage of lexical phenomena, which unavoidably restricts their capabilities and thus the quality of their output. (Of course, shallow lexical semantics is a necessary starting point for a practical system, because it allows for broad coverage.) So, one of the main motivations of the research reported in this thesis was to push the lexical coverage of computational systems to a deeper level.

We remarked in chapter 2 that *polysemy* and *synonymy* are the two fundamental linguistic phenomena that influence the structure of the lexicon. But unlike polysemy, which has been given much attention in linguistics, psychology, lexicography, semantics, and computational linguistics, synonymy has been given relatively little attention. Perhaps this is because synonymy has often been thought of as a 'non-problem'; the argument would go: either there are synonyms, but they are absolute synonyms so not hard to deal with, or there are no synonyms, in which case each word can be handled like any other. But our investigation of *near-synonymy* shows that it is just as complex a phenomenon as polysemy, and that it inherently affects the structure of lexical knowledge. Coming to a greater understanding of near-synonymy was also a motivation of this thesis.

Now, the main practical motivation of this work was to develop a computational process that could "choose the right word" in any situation of language production. In our review of current computational systems that perform *lexical choice* (i.e., MT and NLG systems), we found that no system performed this kind of genuine lexical choice. While major advances have been made in knowledge-based models of the lexicon, the systems are concerned more with structural paraphrasing and a level of semantics allied to syntactic structure. None of the systems captures the fine-grained meanings of and differences between near-synonyms, nor the myriad of choice criteria involved in lexical choice. We also found that statistical models of lexical choice are still in their infancy, but it is clear that their strength will not be in goal-directed choice, but in choosing words that are most typical or natural in context.

Moreover, we found shortcomings in the many theories of lexical semantics, most of which don't even explicitly address near-synonymy. On the one hand, the theories behind the computational models do not account for indirect, fuzzy, or context-dependent meanings. On the other hand, theories that more accurately predict the nature of word meaning (for instance, in cognitive linguistics) are very difficult or impossible to implement in a computational system in their current state.

We decided, therefore, that a new model of lexical knowledge was required that would explicitly account for near-synonymy in a manner implementable in a computational system. To satisfy this need, we developed the *clustered model of lexical knowledge* and a lexical-choice process based on the approximate matching of lexical representations to input representations.

11.2 Contributions of this thesis

11.2.1 Near-synonymy

Our investigation and ensuing classification of near-synonymy into 35 different types is the most comprehensive classification of the phenomenon that we know of. It served as a check list in the development of our model of lexical knowledge, and could be used to compare and evaluate future models with respect to coverage. Yet, the classification is still only a step towards an ultimate classification of the full range of phenomena that would have to be accounted for in the most sophisticated lexicon. For instance, we did not fully investigate context-dependent variation in meaning, or the more complex forms of denotational variation (see section 11.3.1 below). Moreover, the classification is informal because it reflects the kinds of distinction made by a small number of lexicographers. Thus, it should be taken as a starting point to developing a more robust set of categories. For further discussion of the classification as it relates to lexicography, see section 11.3.1 below.

11.2.2 Lexical semantics

In chapter 5, we proposed a preliminary theory of lexical semantics intended to account for near-synonymy and fine-grained lexical meaning. In the theory, the meaning of a word is seen as arising out of a context-dependent combination of a basic inherent context-independent denotation and a set of explicit differences to its near-synonyms. Thus, near-synonyms cluster under a shared denotation and participate in relations of difference that encode their nuances of meaning. We also proposed that word meaning should be represented on three levels, inserting a middle level between the two levels postulated in some current theories (Stede, 1999; Pustejovsky, 1995); the levels are conceptual–semantics, subconceptual / stylistic–semantics, and syntactic–semantics.

Though we did not directly substantiate this theory, we were able to use it as the basis for two complementary models of lexical knowledge, which we showed to be effective in representing many types of lexical nuance.

11.2.3 A statistical model of lexical knowledge

We developed a statistical model of word meaning, or, more accurately, of which of a set of near-synonyms is chosen (or used) in context. In the model, differences between the co-occurrence patterns of the near-synonyms of a cluster are encoded (implicitly) in a lexical co-occurrence network. So, in effect, the model accounts for the *contextual meaning* of a near-synonym on the middle semantic level. (Clusters of near-synonyms were assumed to share an unspecified conceptual meaning.) Then, as a result of an experiment to evaluate the model, we concluded that a purely statistical model was out of reach at this time. To be successful, such a model would require a more elaborate account of local context. It would also need to be trained on a large sense-tagged corpus, which doesn't yet exist. So, although the results are promising and suggest that typicality of usage can be modelled and used in a system for lexical choice, we did not pursue the research any further.

11.2.4 A clustered model of lexical knowledge

We developed a clustered model of lexical knowledge, which is our most significant contribution to lexical semantics. Inspired by the usage notes in synonym discrimination dictionaries (Gove, 1973; Hayakawa, 1994), it is a knowledge-based model that combines aspects of the conventional ontological model of lexical knowledge (Mahesh and Nirenburg, 1995; Stede, 1999; Elhadad, McKeown and Robin, 1997) with aspects of Saussure's paradigmatic view that a word has meaning only because of contrasts to its synonyms (Saussure, 1916). We were able to synthesize these ideas in such a way as to make the model practical for use in an NLP system, thus meeting one of our main objectives. The model keeps the advantages of the conventional model—efficient paraphrasing, lexical choice (at a coarse grain), and mechanisms for reasoning—but overcomes its shortcomings concerning synonymy with its middle level of representation. The model avoids an awkward proliferation of language-dependent concepts in the ontology, and accounts for implications and other nuances of meaning, fuzzy differences between near-synonyms, expressed attitude, and stylistic variation. It also provides a means to choose between near-synonyms, because of its clear and formal representation of differences between near-synonyms.

A cluster is conceived of as a *formal usage note* (DiMarco, Hirst and Stede, 1993) having a *core denotation*; a set of distinctions between near-synonyms in terms of *peripheral concepts* (which act as denotational dimensions), attitudinal dimensions, and stylistic dimensions; and a set of syntactic frames, one for each near-synonym, which encode the compositional and collocational potential of the near-synonyms.

The core denotation of a cluster models the basic language-neutral meaning (on the conceptual–semantic level) shared by the near-synonyms of the cluster. Since we 'cut-off' the ontology at a coarse grain, we simplify what would otherwise be a very complex, even intractable, taxonomy of fine-grained concepts, which in turn facilitates the development of a language-neutral ontology (which is necessary for multilingual applications). Computationally, the core denotation acts as a necessary applicability condition for any of the near-synonyms in the cluster, supporting efficient lexical choice as in the conventional model.

The model enables the representation of differences in near-synonyms by employing a more expressive formalism on the middle level of representation (than on the conceptual level). We formalized the three major types of lexical variation (disregarding collocational variation) each in its own way but within the same overall framework. Each word has a set of lexical distinctions, which distinguish the word from its near-synonyms by, in effect, situating the word relative to its near-synonyms in a multi-dimensional space. So although differences between near-synonyms are encoded implicitly, an explicit difference between two near-synonyms can be computed by comparing the sets of distinctions of the two words. We can also compute the degree of similarity between any two near-synonyms in a cluster.

The model also accounts for cross-linguistic clusters of near-synonyms. In effect, 'language' is treated like any other dimension of variation. This would be most useful in the transfer approach to machine translation.

11.2.5 Lexical choice

Although our clustered model of lexical knowledge is designed to be used in any natural language application, we applied it specifically in lexical choice in text generation (which is also the second half of interlingual machine translation). We first recast lexical choice from the basic problem of concept-to-word mapping into the more general problem of choosing from among alternatives (i.e., near-synonyms) the word that is most appropriate in a particular situation. Then, we noted that both the lexical choice process and the input to the process would necessarily have to be more complex than in current text generation systems, if the fine-grained knowledge, which we can now represent in the lexicon, were to be of any benefit.

We made three contributions in the area of lexical choice. First, realizing that making the 'right choice' might involve tradeoffs between various goals to express different ideas, or might involve the incomplete achievement of other goals, we formalized the input to lexical choice (and the whole sentence planning process) to include both *constraints* and *preferences*. This in itself is not a new result (see (Hovy, 1988; Stede, 1995)). Our contribution is a new formalism for representing preferences that is more extensive than any previous formalism used in text generation. The formalism allows the representation of choice criteria for indirectly expressing semantic content, expressing attitudes towards different entities of a situation, and establishing a style. It can be easily extended to include more choice criteria pertaining to both lexical and syntactic choice.

The second contribution is the two-tiered model of lexical choice, which neatly separates the two major types of decision involved: decisions between clusters and decisions between near-synonyms. Theoretically, a two-tiered process is reasonable since the decisions made on each tier affect the syntactic structure and semantic content of the output in different ways, and so require different processes. In practice, the two-tiered process is a logical consequence of using a clustered model of lexical knowledge. A cluster becomes an option, on the first tier, if its core denotation matches part of the input SitSpec (i.e., the constraints on choice), which in turn makes options, on the second tier, out of all of its near-synonyms. This process is both efficient and robust, ensuring that the right meaning (at a coarse grain) will be lexicalized even if a 'poor' near-synonym were to be chosen in the end.

The third contribution is a solution to the problem of choosing between the near-synonyms of a cluster. Given a set of preferences and a set of near-synonyms that can each satisfy the preferences to different degrees (or possibly not at all), the problem is to find the one that best satisfies the most preferences. Our solution is to use an approximate-matching algorithm, which, for a given word, returns a numeric value that indicates how closely its set of lexical distinctions matches, or satisfies, the set of preferences. For each candidate word, the algorithm determines how well it individually satisfies each preference, combines these values using an overall satisfaction function, and returns the word that maximizes this function.

Finally, in comparing denotational preferences to denotational distinctions, it is sometimes necessary to compute the similarity between two different configurations of concepts. Although we provided only a preliminary solution to this difficult problem, we were able to characterize the problem by distinguishing it from the more general problems of computing lexical similarity and conceptual similarity.

11.2.6 Implementation

We implemented our lexical-choice process within an existing sentence-planning system, MOOSE (Stede, 1999). The resulting prototype system, I-Saurus, can generate and rank a set of sentences, given input consisting of a SitSpec and a set of preferences. We constructed an ontology of about 100 concepts, and represented ten clusters of near-synonyms, which we used to evaluate and demonstrate the capabilities of the model and the system.

I-Saurus, unlike MOOSE, is object-oriented. Every major component, including situations, preferences, clusters, and cluster options, is implemented as a class. This facilitated the implementation of sentence planning and lexical choice and makes many of the components (especially clusters) portable to other applications. During processing, an instantiated situation runs a general lexical-choice process to choose among cluster options, each of which runs a separate lexical-choice process (that can be re-entered to make a different choice should the first choice prove to be unworkable after other choices are made). Each preference individually maintains status about its satisfaction.

MOOSE supplied an architecture for NLG that was intended to be used a testbed for exploring different implementations and theories of each of its components. We successfully used it in this way, and now provide I-Saurus as a similar testbed for further investigating the many factors that influence lexical choice and its relationship to sentence planning.

11.3 Future directions

This research has opened up many avenues for future research. Some of these reflect unresolved issues in the model and system; some involve obvious extensions of the work; and some point to potential applications of the model.

11.3.1 Overcoming the limitations

The influence of context Clearly, many nuances of meaning associated with near-synonyms are context-dependent, so one must take the current context into account when choosing between near-synonyms. But, the clustered model of lexical knowledge does not yet account for how context can influence lexical choice. The model can represent the *possible* influence of context, using the same means as lexicographers use, as observed frequency, but this is obviously a mere descriptive approach. (Note that our statistical model accounts for some aspects of the context, but only insofar as it affects typicality of usage.)

However, our model does not inherently exclude an account of context either. In fact, with its subconceptual level of representation, the model is quite similar to a generative model of the lexicon, even though its focus is on a different phenomenon. Generative models have received much attention recently because of their potential in handling the influence of context on polysemy and sense modulation (Cruse, 1995; Pustejovsky, 1995; Rais-Ghasem, 1998). For instance, Rais-Ghasem proposes a two-level model of word sense structure similar, in a way, to our model for near-synonyms, except that it is designed to account for the related senses of a single word rather than synonyms. His "sense-concepts" encode core characteristics of a word that remain the same in any context, and his "sense-views" describe how a given sense-concept is to be viewed in a particular context, that is, how its senses become activated. Sense-views are defined in part by exemplars, which represent examples of typical use. He defines a mechanism to model sense modulation in text interpretation whereby a particular aspect of a word sense is highlighted depending on the context; e.g., in "the musician moved the *piano*", the heaviness of the *piano* is highlighted rather than its 'musicalness', because *move* co-occurs with heavy objects in its exemplars.

So, if this model were to be superimposed on our model, it would replace our rudimentary use of frequency with a more principled model of how and when different distinctions are conveyed by near-synonyms. In particular, we would like to represent a set of exemplars of the usage of each near-synonym in a cluster. Then by treating the distinctions as though they were sense-views, we could use the above mechanism (modified, of course) to determine, say, whether the word *blunder* expresses blameworthiness or stupidity, or whatever else, in a context such as:

11.1 They can hardly restrain themselves from raising the question of whether Republicans, if they had been in power, would have made "amateurish and monumental *blunders*" in Cuba.[1]

For instance, from the lexical representation of *blunder* we can determine that it sometimes expresses blameworthiness, but perhaps we can we say so definitively in this particular context, because of the use of certain other structures that are present in exemplars of the usage of *blunder*. For instance, the structure "raising the question" might be an indicator of blameworthiness. Of course, this presumes that the right kind of knowledge has been represented in the lexicon, which is a significant task in its own right.

Lexical acquisition and lexicography If our models are ever to be successful, both a large ontology and a large number of clusters will have to be acquired and represented. Acquiring the knowledge required by our statistical model is relatively easy once a proper corpus has been assembled, but in our clustered model, lexical-knowledge acquisition is a major bottleneck for development, as it is in other knowledge-based systems. Work on constructing large ontologies for computational linguistics is proceeding slowly, as we remarked in chapter 4, and many techniques for automatic lexical acquisition have also been developed (Zernik, 1991), but not for the type of fine-grained knowledge that we need to represent. One could proceed in three directions. First, the best way to get this information would be to automatically convert the data in the existing synonym discrimination dictionaries—if we could gain access to machine-readable versions of them—into the proper format for our model. We began to develop a method for doing this in chapter 3, but since we had to carry it out by hand, there was no way to evaluate its effectiveness. Second, one could go straight to the lexicographers with a special lexicographical workbench that would help lexicographers to write computational entries rather than textual entries. Such tools already exist for the development of regular dictionaries, so it is possible that one could be modified for the present purpose. Finally, one could devise better methods to automatically extract the required knowledge from large text corpora and the currently available MRDs. The ultimate goal is to construct a computational lexical database on the scale of WordNet (Fellbaum, 1998).

But before this endeavour can proceed at this scale, we must develop a more formal and robust classification of lexical variation. Because, although we saw agreement between the usage notes for the same set of near-synonyms prepared by different lexicographers (see section 3.5.3), it is not at all assured that two lexicographers will come to the same conclusions in any given case, let alone the same encoding into a formal representation. Thus, we must define a set of objective diagnostics for each type of lexical variation so that anyone can determine that, for instance, *error* differs from *blunder* in the *implication* of criticism rather than in the *emphasis* of

[1] *Brown Corpus.*

criticism. The first step in this project will be to use the current classification to see if diagnostics can indeed be defined, and to refine the classification as appropriate. Of course, changes to the classification will not affect the clustered model of lexical knowledge, only the knowledge that is encoded in the model.

Evaluation As we discussed in chapter 10, evaluating NLP systems is very difficult. In evaluating our system, we identified a number of test cases aimed at demonstrating the coverage of the model, but this was only a first step. We must develop a large suite of test materials specially crafted for evaluating sophisticated lexical choice. For instance, it should contain a representative set of sample sentences in which the right and wrong options (or a ranked set of options) is specified for every word that has near-synonyms. But since judging the quality of lexical choice is so subjective, and many people might not even be consciously aware of the subtle differences in word meaning, this methodology might not always be effective. We could instead use a methodology in which subjects are asked, after comparing two sentences (or larger texts), which of several particular differences is most evident to them (e.g., “is text *A* more formal than text *B*?”). We could then determine if their judgments of the texts match what the system was attempting to express. Clearly, this is a very underdeveloped area of research.

11.3.2 Developing the models further

Cross-linguistic clusters and near-synonymy Our theory of near-synonymy assumes that near-synonyms share a common denotation even across languages (see section 7.1). This might not always be the case—after all, if two different languages can each distinguish corresponding clusters of near-synonyms in different ways, it follows that their core denotations might also be slightly different. If this is so, then it mitigates against a direct transfer approach to machine translation, since some clusters would not align across languages.

The real problem is our definition of near-synonymy, which implies that near-synonyms must share a common (essential) denotation. This definition, on purpose, glossed over both the contextual nature of near-synonymy and the prototypicality effects that most words would seem to exhibit. It forces synonym clusters to be static, when in reality they must have gradations of membership and no fixed boundaries.

One way to improve the clustered model would be to investigate a method of comparing different clusters by using a context-dependent similarity function that would take into account both the core denotations and the peripheral concepts of the clusters. If two clusters (in the same or different languages) were found to be similar enough in a particular context, then they could be temporarily merged.

Polysemy and vagueness Another way to approach the above problem of fuzzy clusters is to draw on the research on semantic vagueness (Tuggy, 1993; Cruse, 1995). Tuggy argues that there is an ambiguity–vagueness cline with polysemy in the middle. An ambiguous word has two (or more) senses that are clearly distinct (e.g., *bank* ‘financial institution’ vs. *bank* ‘land at river edge’). A vague word has two (or more) non-distinguished subcases of a more general sense (e.g., *paint* ‘paint a portrait’ vs. *paint* ‘paint a landscape’). What interests us here is the similarity between near-synonymy and semantic vagueness. For instance, it seems that the closer that two senses of a vague word are related, the more they resemble near-synonyms, albeit with the same phonological form. For example, if the sense of the word *blunder* actually had the phonological form *error*, then *error* would be vague, because of its other sense. Crosslinguistically, one can find many similar examples; for instance, *river* can be considered vague in English, because in French the same idea is covered by two words, *rivière* and *fleuve*. Viegas (1998) describes a computational model of lexical knowledge that begins to deal with this phenomenon, but the model does not explicitly recognize near-synonymy.

Merging the statistical model and the knowledge-based model We would now like to merge our two models into a hybrid statistical / knowledge-based model, because we believe that such a model is required to fully account for near-synonymy and sophisticated lexical choice. A statistical model can encode a broad but shallow coverage of knowledge about word usage in context, so it is an effective way of representing the contextual meaning of a word. A knowledge-based model, on the other hand, is required for goal-directed lexical choice. Not only would a hybrid model be more robust, but it would also be better able to cope with the influence of context on lexical choice. For instance, we claim that it would be able to handle the phenomenon of *evoked meaning* (Cruse, 1986). In a preliminary computational account of evoked meaning, Edmonds (1996) proposed that if one chooses a near-synonym that is not typical in a particular context (which we can determine with our statistical model), it would evoke a meaning in that context. That is, it would emphasize those aspects of the meaning of the chosen word that were different from the meaning of the typical word (which can be determined from our clustered model).

Psycholinguistic experiments As we said in section 5.6, we need to determine what kinds of psycholinguistic predictions our theory of lexical meaning makes, and we must empirically test these predictions. As we said, one prediction that the theory makes is that it should be more difficult for people to articulate differences between near-synonyms than between words that are merely similar, because the differences are represented subconceptually and are thus not structurally alignable (Markman and Gentner, 1993). Hirst and Edmonds tested this prediction by repeating one of Markman and Gentner's experiments, this time including near-synonyms in the stimulus sets. However, as we discussed in section 5.6, the results of the experiment were inconclusive. More research is required to determine if the experimental design was flawed—perhaps the design is inappropriate for near-synonyms—or if different sets of stimuli were instead required. More likely, we will have to form an alternate hypothesis, perhaps involving a finer grained typology of differentiae that accounts for more than the simple alignable versus non-alignable distinction. For instance, do subjects tend to list more stylistic differences for near-synonyms? Or more ontologically specific, or general, types of difference?

Criteria for lexical choice We formalized a variety of lexical-choice criteria, but there are many criteria left to formalize including emphasizing an aspect of the situation, making a participant of the situation salient, expressing emotion, speaking in a certain dialect, using morphologically appropriate words (perhaps for rhyming), and avoiding repetitious uses of words. We must also incorporate collocational constraints into the model. And of the criteria that we did formalize (i.e., denotational, attitudinal, and stylistic criteria), we must improve our formalism. For example, we need to model the more complex types of denotational distinctions and the logical relationships between them. We discussed the shortcomings of the model in this respect in section 7.5.

To improve the lexical-choice process, research is required on handling the tradeoffs in the satisfaction of preferences and the partial satisfaction of preferences, and in avoiding unwanted implicature. So, we must investigate how to manage all of the interacting constraints and preferences on lexical choice. Crucially, we must improve our very preliminary measures for near-synonymic similarity and conceptual similarity. As we said in chapter 8, the solution lies in part in representing (in the ontology or elsewhere) the relationships between the various denotational aspects of the near-synonyms in a cluster.

Pragmatics and computational stylistics The lexicon is but an interface in a larger system. Our model might allow a system to express a meaning indirectly or to establish a certain style by choosing the appropriate words, but it does not model what the system can achieve (pragmatically) by doing this, or why the system would want to use a particular means to achieve a particular communicative goal in the first place. Such decisions are assumed to have already been made by another process that supplies the input to our system. In machine translation, we would have to recover the nuances from the source text and provide them as input to our system (see below). But in text generation, the source of the input is less clear. To take advantage of the kind of knowledge that can be represented in our lexicon, we must augment current text-planning systems to at least use ‘rules’ such as “if you have to express a negative idea, do so indirectly”, and “to impress someone, use formal style”. Hovy (1988) was the first to discuss these issues, and since then, a significant amount of research has been done on pragmatics and stylistics, but it has been difficult to apply the work in real systems because the sentence-planning components were not expressive enough. Our clustered model now provides an ideal opportunity to carry forward research on the relationship between pragmatics and lexical semantics.

11.3.3 Applications

Sentence planning We followed Stede’s MOOSE, and formalized sentence planning as a search problem. Stede concluded that research must be done on the very intriguing problem of managing the possibly intractable search process; our new lexical-choice process makes this even more urgent. Right now, an exhaustive search is avoided by using a heuristic that forces the algorithm to make the most preferred local decisions, but this assumes that local decisions do not affect each other, which is a gross simplification. We must do more research into the relationship between lexical decisions and syntactic decisions, so that we can find a better way to manage the search space. For instance, if we know that a whole cluster will satisfy very few of the preferences, then it should be ignored. And if certain words in a cluster are obvious poor choices, not satisfying any preferences, then they should also be skipped.

Perhaps formulating sentence planning as searching is wrong to begin with. One might consider a kind of distributed approach in which each potential node of a sentence plan simultaneously vies for selection in the final plan. In this strategy, every cluster option would simultaneously put forward its current best near-synonym, and either the clusters would somehow come to mutual agreement on the best set of options (perhaps using a spreading activation algorithm), or a mediator would decide which options were collectively the best. This strategy takes the blackboard approach to the limit where many *instantiations* of modules compete rather than just individual modules (cf. Wanner and Hovy’s (1996) approach).

Recovering nuances from text The clustered model is designed to be used in the analysis stage of an MT system as well. One very interesting and completely open area of research here is in recovering nuances from text. Since many nuances are expressed indirectly and are influenced by the context, one cannot know for sure whether they have been expressed unless one performs a very thorough analysis. So, it might not be possible for even a reasonably thorough analysis to decide whether a nuance was expressed, or how indirectly it was expressed, given the context-dependent nature of word meaning. But, on the basis of the knowledge of what words can express, which can be represented in the clustered model, the analysis stage of MT might be able to determine at least some *possibilities* of what is expressed in a source text. Then, depending on the type of MT system, the appropriate target language words could be chosen. In a transfer-based system, we would have to directly compare the lexical representations of near-synonyms in different languages while factoring in the possibilities. In an interlingual system, we could build an interlingual representation that includes the possibilities, which would become preferences for generation. Edmonds (1998) further develops these ideas.

An intelligent thesaurus Current electronic dictionaries and thesauri are really no more sophisticated than their paper counterparts. But we envision an interactive electronic dictionary—an *intelligent thesaurus*—that could actively help a person to find and choose the right word in any particular context. Rather than merely list the possibilities, it would rank them according to the context and to parameters supplied by the user. It would also help a person to understand the subtleties of a text (in a native or foreign language). So, an intelligent thesaurus could also be very useful in computer-assisted second language instruction. We believe that our contributions are an excellent base for research in this area.

Automatic editing of text There is also a need for the automatic editing of text, say, to make a text conform to a certain stylistic standard, to change word uses that have nuances that might be misunderstood, or to make a text more readable, natural, or simple. For instance, Devlin and Tait (1998) report on a “lexical simplification” system that is intended to simplify texts to make them easier for aphasics to read. Their current implementation simply replaces low frequency words with high frequency words, but it’s not hard to see that it could be improved if it were to use a model of typical usage (like our statistical model) or a knowledge-based model to replace words on other grounds as well.

Pause . . .

Bibliography

Allan, Keith and Burridge, Kate (1991). *Euphemism and Dysphemism: Language Used as Shield and Weapon*. Oxford University Press, New York.

Arnold, Doug; Sadler, Louisa; and Humphreys, R. Lee (1993). Evaluation: An assessment. *Machine Translation*, 8(1–2):1–24. (Introduction to the special issue on the evaluation of MT systems.).

Baader, Franz; Borgida, Alex; and McGuinness, Deborah L. (1998). Matching in description logics: Preliminary results. In *International Workshop on Description Logics (DL98)*, Trento, Italy.

Bailly, René (1947). *Dictionnaire des synonymes de la langue française*. Librairie Larousse.

Barrière, Caroline (1997). *From a Children's First Dictionary to a Lexical Knowledge Base of Conceptual Graphs*. Ph.D. thesis, Simon Fraser University, Vancouver, Canada.

Barsalou, Lawrence W. (1992). Frames, concepts, and conceptual fields. In (Lehrer and Kittay, 1992a), pages 21–74.

Batchelor, Ronald E. and Offord, Malcolm H. (1993). *Using French Synonyms*. Cambridge University Press.

Bateman, John A. (1996). KPML development environment. Technical report, IPSI, GMD, Darmstadt, Germany.

Bateman, John A. and Paris, Cécile L. (1991). Constraining the deployment of lexicogrammatical resources during text generation: Towards a computational instantiation of register theory. In Ventola, Eija, editor, *Functional and Systemic Linguistics: Approaches and Uses*, Trends in linguistics. Studies and monographs. 55, pages 81–106. Mouton de Gruyter, New York.

Beale, Stephen; Nirenburg, Sergei; Viegas, Evelyne; and Wanner, Leo (1998). De-constraining text generation. In *Proceedings of the Ninth International Workshop on Natural Language Generation*, pages 48–57.

Bénac, Henri (1956). *Dictionnaire des synonymes*. Paris Hachette.

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*. John Benjamins, Amsterdam.

Biber, Douglas (1988). *Variation Across Speech and Writing*. Cambridge University Press.

Brachman, Ronald and Schmolze, J. (1985). An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2).

Brown, Peter F.; Cooke, John; Della Pietra, Stephen A.; Della Pietra, Vincent J.; Jelinek, Frederick; Lafferty, John D.; Mercer, Robert L.; and Roossin, Paul S. (1990). A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Brown, Peter F.; Della Pietra, Stephen A.; Della Pietra, Vincent J.; and Mercer, Robert L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–312.

Budanitsky, Alexander (forthcoming, 1999). *Measuring Semantic Relatedness and its Applications*. Master's thesis, University of Toronto, Toronto, Canada.

Chapman, Robert L., editor (1992). *Roget's International Thesaurus*. HarperCollins Publishers, 5th edition.

Church, Kenneth Ward; Gale, William; Hanks, Patrick; and Hindle, Donald (1991). Using statistics in lexical analysis. In (Zernik, 1991), pages 115–164.

Church, Kenneth Ward; Gale, William; Hanks, Patrick; Hindle, Donald; and Moon, Rosamund (1994). Lexical substitutability. In Atkins, B.T.S. and Zampolli, A., editors, *Computational Approaches to the Lexicon*, pages 153–177. Oxford University Press.

Clark, Eve V. (1992). Conventionality and contrast: Pragmatic principles with lexical consequences. In (Lehrer and Kittay, 1992a), pages 171–188.

Cohen, William W.; Borgida, Alex; and Hirsh, Haym (1992). Computing least common subsumers in description logic. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, pages 754–760.

Coseriu, E. (1967). Lexikalische Solidaritäten. *Poetica*, 1:293–303.

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

Cruse, D. Alan (1995). Polysemy and related phenomena from a cognitive linguistic viewpoint. In (Saint-Dizier and Viegas, 1995), pages 33–49.

Dagan, Ido; Marcus, Shaul; and Markovitz, Shaul (1993). Contextual word similarity and estimation from sparse data. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 164–171.

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

Davitz, Joel (1969). *The Language of Emotion*. Academic Press.

Devlin, Siobhan and Tait, John (1998). The use of a psycholinguistic database in the simplification of text for aphasic readers. In Nerbonne, John, editor, *Linguistic Databases*, pages 161–173. CSLI Publications, Stanford, CA.

DiMarco, Chrysanne (1994). The nature of near-synonymic relations. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, pages 691–695, Kyoto, Japan.

DiMarco, Chrysanne and Hirst, Graeme (1993). A computational theory of goal-directed style in syntax. *Computational Linguistics*, 19(3):451–500.

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

DiMarco, Chrysanne; Hirst, Graeme; Wanner, Leo; and Wilkinson, John (1995). Healthdoc: Customizing patient information and health education by medical condition and personal characteristics. In *Workshop on Artificial Intelligence in Patient Education*, Glasgow.

Dorr, Bonnie J. (1993). *Machine Translation: A View from the Lexicon*. MIT Press, Cambridge, Mass.

Dorr, Bonnie J. (1994). Machine translation divergences: A formal description and proposed solution. *Computational Linguistics*, 20(4):597–634.

Dubois, Didier and Prade, Henri (1980). *Fuzzy Sets and Systems: Theory and Applications*. Academic Press, New York.

Dubois, Didier and Prade, Henri (1988). *Possibility Theory: An Approach to Computerized Processsing of Uncertainty*. Plenum Press, New York.

Edmonds, Philip (1996). Evoking meaning by choosing the right words. In *Proceedings of the First Student Conference in Computational Linguistics in Montreal*, pages 80–87.

Edmonds, Philip (1997). Choosing the word most typical in context using a lexical co-occurrence network. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 507–509, Madrid, Spain.

Edmonds, Philip (1998). Translating near-synonyms: Possibilities and preferences in the interlingua. In *Proceedings of the AMTA/SIG-IL Second Workshop on Interlinguas*, pages 23–30, Langhorne, PA. (Proceedings published as technical report MCCS-98-316, Computing Research Laboratory, New Mexico State University.).

Egan, Rose F. (1973). Survey of the history of English synonymy. In Gove, Philip B., editor, *Webster's New Dictionary of Synonyms*, pages 5a–31a. G. & C. Merriam Co.

Elhadad, Michael; McKeown, Kathleen; and Robin, Jacques (1997). Floating constraints in lexical choice. *Computational Linguistics*, 2(23):195–240.

Elhadad, Michael and Robin, Jacques (1996). An overview of surge: A reusable comprehensive syntactic realization component. Technical Report 96-03, Ben Gurion University, Dept. of Computer Science, Beer Sheva, Israel.

Evens, Martha, editor (1988). *Relational Models of the Lexicon: Representing Knowledge in Semantic Networks*. Cambridge University Press.

Farrell, Ralph Barstow (1977). *Dictionary of German Synonyms*. Cambridge University Press, third edition.

Fellbaum, Christiane, editor (1998). *WordNet: An Electronic Lexical Database*. MIT Press.

Fernald, James C., editor (1947). *Funk & Wagnall's Standard Handbook of Synonyms, Antonyms, and Prepositions.* Funk & Wagnall's Company, Inc., New York.

Fillmore, Charles J. (1982). Frame semantics. In of Korea, The Linguistic Society, editor, *Linguistics in the Morning Calm: Selected Papers from SICOL-1981*, pages 111–137. Hanshin Publishing Company, Seoul.

Firth, J. R. (1957). A synopsis of linguistic theory, 1930–55. In Philological Society, London, editor, *Studies in Linguistic Analysis*, pages 1–31. Oxford Blackwell. Reprinted in Palmer, F. R. (ed.), *Selected Papers of J. R. Firth 1952–59.* pp. 168–205. Longmans, Green, and Co. Ltd., London, 1968.

Frege, Gottlob (1892). Über Sinn und Bedeutung. *Zeitscrift fur Philosophie und Philosophie Kritik*, 100:25–50. English translation: On sense and reference. In Geach, P. & Black M., editors, *Translations from the Philosophical Writings of Gottlob Frege.* Blackwell, Oxford, 1960.

Gale, William; Church, Ken; and Yarowsky, David (1992). A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26:415–439.

Gale, William A. and Church, Kenneth W. (1993). A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102.

Geeraerts, Dirk; Grondelaers, Stefan; and Bakema, Peter (1994). *The Structure of Lexical Variation: Meaning, Naming, and Context.* Mouton de Gruyter, New York.

Gentner, Dedre and Markman, Arthur B. (1994). Structural alignment in comparison: No difference without similarity. *Psychological Science*, 5(3):152–158.

Goddard, Cliff (1994). Semantic theory and lexical universals. In Goddard, Cliff and Wierzbicka, Anna, editors, *Semantic and Lexical Universals*, pages 7–29. John Benjamins, Philadelphia.

Golding, Andrew R. and Schabes, Yves (1996). Combining trigram-based and feature-based methods for context-sensitive spelling correction. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics.*

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

Goodman, N. (1952). On likeness of meaning. In Linsky, L., editor, *Semantics and the Philosophy of Language.* University of Illinois Press, Urbana, Ill.

Gove, Philip B., editor (1973). *Webster's New Dictionary of Synonyms.* G. & C. Merriam Co.

Grefenstette, Gregory (1994). *Explorations in Automatic Thesaurus Discovery.* Kluwer Academic Publishers.

Halliday, M. A. K. (1961). Categories of the theory of grammar. *Word*, 17:241–292.

Halliday, M. A. K. (1978). *Language as a Social Semiotic: The Social Interpretation of Language and Meaning.* Edward Arnold, London.

Halliday, M. A. K. (1994). *An Introduction to Functional Grammar.* Edward Arnold, London, 2nd edition.

Hasan, Ruqaiya (1987). The grammarian's dream: Lexis as most delicate grammar. In Halliday, M. A. K. and Fawcett, Robin P., editors, *New Developments in Systemic Linguistics*, pages 184–211. Pinter, New York.

Hausmann, F. (1985). Kollokationen im Deutschen Woerterbuch: ein Beitrag zur Theorie des lexicographischen Biespiels. In Bergenholtz, H. and Mugdon, J., editors, *Lexikographie und Grammatik*. Niemeyer, Turgen, Germany.

Hayakawa, S. I., editor (1994). *Choose the Right Word: A Contemporary Guide to Selecting the Precise Word for Every Situation*. HarperCollins Publishers, New York.

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*, pages 51–56, Stanford, CA.

Horacek, Helmut (1990). The architecture of a generation component in a complete natural language dialogue system. In Dale, Robert; Mellish, Chris; and Zock, Micheal, editors, *Current Research in Natural Language Generation*. Academic Press, London.

Horacek, Helmut and Zock, Michael, editors (1993). *New Concepts in Natural Language Generation: Planning, Realization, and Systems*. Pinter Publishers, London.

Hovy, Eduard (1988). *Generating Natural Language Under Pragmatic Constraints*. Lawrence Erlbaum Associates.

Hutchins, W. John and Somers, Harold L. (1992). *An Introduction to Machine Translation*. Academic Press.

Iordanskaja, Lidija; Kittredge, Richard; and Polguère, Alain (1991). Lexical selection and paraphrase in a meaning-text generation model. In (Paris, Swartout and Mann, 1991), pages 293–312.

Jackendoff, Ray (1983). *Semantics and Cognition*. MIT Press.

Jackendoff, Ray (1990). *Semantic Structures*. MIT Press, Cambridge, Mass.

Jiang, Jay J. and Conrath, David W. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the International Conference for Research on Computational Linguistics (ROCLING X)*, Taiwan.

Karov, Yael and Edelman, Shimon (1998). Similarity-based word sense disambiguation. *Computational Linguistics*, 24(1):41–60.

Kasper, Bob (1989). A flexible interface to linking applications to Penman's sentence generator. In *Proceedings of the 1989 DARPA Speech and Natural Language Workshop*, pages 153–158.

Katz, Jerrold J. and Fodor, J. A. (1963). The structure of a semantic theory. *Language*, 39:170–210.

Kay, Martin and Röscheisen, Martin (1993). Text-translation alignment. *Computational Linguistics*, 19(1):121–142.

Kay, Mairé Weir, editor (1988). *Webster's Collegiate Thesaurus*. Merriam-Webster, Springfield, Mass.

Knight, Kevin; Chander, Ishwar; Haines, Matthew; Hatzivassiloglou, Vasileios; Hovy, Eduard; Iida, Masayo; Luk, Steve K.; Whitney, Richard; and Yamada, Kenji (1995). Filling knowledge gaps in a broad-coverage machine translation system. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 1390–1396, Montreal.

Knight, Kevin and Hatzivassiloglou, Vasileios (1995). Two-level, many-paths generation. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 252–260.

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.

Kozima, Hideki and Furugori, Teiji (1993). Similarity between words computed by spreading activation on an English dictionary. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, pages 232–239, Utrecht.

Kumano, Akira and Hirakawa, Hideki (1994). Building an MT dictionary from parallel texts based on linguistic and statistical information. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, pages 76–81, Kyoto, Japan.

Lakoff, George (1987). *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*. University of Chicago Press, Chicago.

Langenscheidt (1995). *Langenscheidt's New College German Dictionary: German–English, English–German*. Langenscheidt, Berlin.

Langkilde, Irene and Knight, Kevin (1998). The practical value of n-grams in generation. In *Proceedings of the Ninth International Workshop on Natural Language Generation*, pages 248–255, Niagara-on-the-Lake, Canada.

Leacock, Claudia and Chodorow, Martin (1998). Combining local context with WordNet similarity for word sense identification. In (Fellbaum, 1998).

Lehrberger, John and Bourbeau, Laurent (1988). *Machine Translation: Linguistic Characteristics of MT Systems and General Methodology of Evaluation*. John Benjamins.

Lehrer, Adrienne (1974). *Semantic Fields and Lexical Structure*. North Holland, Amsterdam.

Lehrer, Adrienne and Kittay, Eva Feder, editors (1992a). *Frames, Fields, and Contrasts: New Essays in Semantic and Lexical Organization*. Lawrence Erlbaum Associates.

Lehrer, Adrienne and Kittay, Eva Feder (1992b). Introduction. In (Lehrer and Kittay, 1992a), pages 1–20.

Lenat, D. B. (1995). CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11).

Levelt, Willem J. M. (1989). *Speaking: From Intention to Articulation*. MIT Press, Cambridge, Mass.

Li, Deyi and Liu, Dongbo (1990). *A Fuzzy PROLOG Database System*. Wiley, New York.

Lin, Dekang (1998). Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (COLING-ACL-98)*, pages 768–774, Montreal.

Lyons, John (1977). *Semantics*. Cambridge University Press, Cambridge.

Lyons, John (1981). *Language, Meaning and Context*. Fontana, London.

Lyons, John (1995). *Linguistic Semantics: An Introduction*. Cambridge University Press, Cambridge.

MacGregor, R. and Bates, R. (1987). The Loom knowledge representation language. Technical Report ISI/RS-87-188, USC Information Sciences Institute.

Mahesh, Kavi and Nirenburg, Sergei (1995). A situated ontology for practical NLP. In *Proceedings of the IJCAI-95 Workshop on Basic Ontological Issues in Knowledge Sharing*, Montreal.

Markman, Arthur B. and Gentner, Dedre (1993). Splitting the differences: A structural alignment view of similarity. *Journal of Memory and Language*, 32:517–535.

Martin, J. R. (1992). *English Text: System and Structure*. John Benjamins.

Matthiessen, Christian M. I. M. (1991). Lexico(grammatical) choice in text generation. In (Paris, Swartout and Mann, 1991), pages 249–292.

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, B. J., Sparck Jones, K., and Webber B. L. (eds.), *Readings in Natural Language Processing*. Morgan Kaufmann Publishers, 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.

McDonald, David D. (1991). On the place of words in the generation process. In (Paris, Swartout and Mann, 1991), pages 227–247.

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

McMahon, John G. and Smith, Francis J. (1996). Improving statistical language model performance with automatically generated word hierarchies. *Computational Linguistics*, 22(2):217–248.

Melamed, I. Dan (1997). A word-to-word model of translational equivalence. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 490–497, Madrid, Spain.

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 Wanner, Leo (1994). Lexical co-occurrence and lexical inheritance. Emotion lexemes in German: A lexicographic case study. In *Lexikos 4*, Afrilex-Reeks 4. Stellenbosch RSA.

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.

Miller, George; Beckwith, R.; Fellbaum, C.; Gross, D.; and Miller, K. (1990). Five papers on WordNet. Technical Report CSL Report 43, Princeton University.

Morris, Jane and Hirst, Graeme (1991). Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1):21–48.

Ng, Hwee Tou and Lee, Hian Beng (1996). Integrating multiple sources to disambiguate word sense: An exemplar-based approach. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*.

Nicolov, Nicolas; Mellish, Chris; and Ritchie, Graeme (1996). Approximate generation from non-hierarchical representations. In *Proceedings of the Eighth International Workshop on Natural Language Generation*, pages 31–40.

Nirenburg, Sergei, editor (1995). The Pangloss Mark III machine translation system. Technical Report CMU-CMT-95-145, Carnegie Mellon University, Center for Machine Translation. Jointly issued by Computing Research Laboratory (New Mexico State University) and Information Sciences Institute (University of Southern California).

Nirenburg, Sergei; Carbonell, Jaime; 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 Levin, Lori (1992). Syntax-driven and ontology-driven lexical semantics. In (Pustejovsky and Bergler, 1992), pages 5–20.

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.

Nirenburg, Sergei; Raskin, Victor; and Onyshkevych, Boyan (1995). Apologiae ontologiae. In *Proceedings of the Conference on Theoretical Issues in Machine Translation*, Leuven, Belgium.

Nogier, Jean-François and Zock, Michael (1992). Lexical choice as pattern matching. *Knowledge-Based Systems*, 5:200–212.

Nyberg, Eric H.; Mitamura, Teruko; and Carbonell, Jaime (1994). Evaluation metrics for knowledge-based machine translation. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, pages 95–99.

Panaget, Franck (1994). Using a textual representational level component in the context of discourse or dialogue generation. In *Proceedings of the Seventh International Workshop on Natural Language Generation*, pages 127–136, Kennebunkport, Maine.

Paris, Cécile L.; Swartout, William R.; and Mann, William C., editors (1991). *Natural Language Generation in Artificial Intelligence and Computational Linguistics*. Kluwer Academic, Boston.

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

Pereira, Fernando; Tishby, Naftali; and Lee, Lillian (1993). Distributional clustering of English words. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 183–190.

Pustejovsky, James (1995). *The Generative Lexicon*. MIT Press, Cambridge, Mass.

Pustejovsky, James and Bergler, Sabine, editors (1992). *Lexical Semantics and Knowledge Representation: First SIGLEX Workshop*. Lecture Notes in Artificial Intelligence 627. Springer-Verlag.

Pustejovsky, James and Boguraev, Branimir, editors (1996). *Lexical Semantics: The Problem of Polysemy*. Clarendon Press, New York.

Quine, W. V. (1951). Two dogmas of empiricism. *Philosophical Review*, 60:20–43.

Rais-Ghasem, Mohsen (1998). *An Exemplar-Based Account of Contextual Effects*. Ph.D. thesis, Carleton University, Ottawa, Canada.

Rambow, Owen and Korelsky, Tanya (1992). Applied text generation. In *Proceedings of the Third Conference on Applied Natural Language Processing (ANLP-1992)*, pages 40–47.

Randall, Bernice (1991). *When is a Pig a Hog: A Guide to Confoundingly Related English Words*. Prentice Hall.

Reiter, Ehud (1991). A new model of lexical choice for nouns. *Computational Intelligence*, 7(4):240–251.

Reiter, Ehud (1994). Has a consensus NL generation architecture appeared, and is it psycholinguistically plausible? In *Proceedings of the Seventh International Workshop on Natural Language Generation*, pages 163–170, Kennebunkport, Maine.

Reiter, Ehud and Dale, Robert (1997). Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–88.

Resnik, Philip (1995). Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453, Montreal.

Resnik, Philip S. (1993). *Selection and Information: A Class-based Approach to Lexical Relationships*. Ph.D. thesis, University of Pennsylvania.

Richardson, Stephen D. (1997). *Determining Similarity and Inferring Relations in a Lexical Knowledge Base*. Ph.D. thesis, City University of New York, New York City.

Robin, Jacques (1990). Lexical choice in natural language generation. Technical Report CUCS-040-90, Columbia University.

Room, Adrian (1981). *Room's Dictionary of Distinguishables*. Routledge & Kegan Paul, Boston.

Rosch, Eleanor (1978). Principles of categorization. In Rosch, Eleanor and Lloyd, Barbara B., editors, *Cognition and categorization*, pages 27–48. Lawrence Erlbaum Associates.

Ruhl, Charles (1989). *On Monosemy: A Study in Linguistic Semantics*. State University of New York Press, Albany.

Saint-Dizier, Patrick and Viegas, Evelyn, editors (1995). *Computational Lexical Semantics*. Cambridge University Press, Cambridge, England.

Saussure, Ferdinand de (1916). *Cours de linguistique générale*. Translated by Roy Harris as *Course in general linguistics*, London: G. Duckworth, 1983.

Schank, Roger C. (1975). *Conceptual Information Processing*. North-Holland, Amsterdam.

Schütze, Hinrich (1998). Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

Sinclair, John McH. (1966). Beginning the study of lexis. In Bazell, C. E.; Catford, J. C.; Halliday, M. A. K.; and Robins, R. H., editors, *In Memory of J. R. Firth*, pages 410–430. Longmans, Green, and Co. Ltd., London.

Smadja, Frank (1991). From N-grams to collocations, an evaluation of Xtract. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 279–285.

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

Smadja, Frank; McKeown, Kathleen; and Hatzivassiloglou, Vasileios (1996). Translating collocations for bilingual lexicons: A statistical approach. *Computational Linguistics*, 22(1):1–38.

Smith, Edward E. and Medin, Douglas L. (1981). *Categories and Concepts*. Harvard University Press, Cambridge, MA.

Sowa, John F. (1984). *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, New York.

Sparck Jones, Karen (1986). *Synonymy and Semantic Classification*. Edinburgh University Press.

Sparck Jones, Karen and Galliers, Julia R. (1996). *Evaluating Natural Language Processing Systems: An Analysis and Review*. Springer-Verlag, Berlin.

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 (1995). Lexicalization in natural language generation: A survey. *Artificial Intelligence Review*, 8:309–336.

Stede, Manfred (1996). Lexical paraphrases in multilingual sentence generation. *Machine Translation*, 11:75–107.

Stede, Manfred (1998). A generative perspective on verb alternations. *Computational Linguistics*, 24(3):401–430.

Stede, Manfred (1999). *Lexical Semantics and Knowledge Representation in Multilingual Text Generation*. Kluwer Academic Publishers, Boston / Dordrecht / London.

Summers, Della, editor (1993). *Longman Language Activator*. Longman, Harlow, England.

Tarski, A. (1944). The semantic conception of truth. *Philosophy and Phenomenological Research*, 4:341–375.

Taylor, John R. (1989). *Linguistic Categorization: Prototypes in Linguistic Theory*. Clarendon Press, London.

Taylor, John R. (1997). Synonymy. Unpublished manuscript.

Toglia, Michael P. and Battig, William F. (1978). *Handbook of Semantic Word Norms*. Lawrence Erlbaum Associates, Hillsdale.

Trier, Jost (1934). Das Sprachliche Feld. Eine Auseinandersetzung. *Neue Jahrbücher für Wissenschaft und Jugendbildung*, 10:428–449.

Tucker, Gordon H. (1995). *The Treatment of Lexis in a Systemic Functional Model of English with Special Reference to Adjectives and their Structure*. Ph.D. thesis, School of English Studies, Communications and Philosophy, University of Wales College of Cardiff.

Tuggy, David (1993). Ambiguity, polysemy, and vagueness. *Cognitive Linguistics*, 4(3):273–290.

Ullmann, Stephen (1962). *Semantics: An Introduction to the Science of Meaning*. Blackwell, Oxford.

Viegas, Evelyne (1998). Multilingual computational semantic lexicons in action: The WYSINNWYG approach to NLG. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (COLING-ACL-98)*, pages 1321–1327, Montreal, Canada.

Vossen, Piek (1998). *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*. Kluwer Academic Publishers.

Vossen, Piek; Díez-Orzas, Pedro; and Peters, Wim (1997). The multilingual design of the EuroWordNet database. In *IJCAI-97 Workshop on Multilingual Ontologies for NLP Applications*, Nagoya, Japan.

Vossen, Piek; Peters, Wim; and Gonzalos, Julio (1999). Towards a universal index of meaning. In *Proceedings of SIGLEX99: Standardizing Lexical Resources*, Maryland, USA.

Wanner, Leo and Hovy, Eduard (1996). The HealthDoc sentence planner. In *Proceedings of the Eighth International Workshop on Natural Language Generation*, pages 1–10.

Ward, Nigel (1994). *A Connectionist Language Generator*. Ablex Publishing, New Jersey.

Whitelock, Pete (1994). Shake-and-bake translation. In Rupp, C. J.; Rosner, M. A.; and Johnson, R. L., editors, *Constraints, Language, and Computation*, pages 339–359. Academic Press, London.

Whitten, William B.; Suter, W. Newton; and Frank, Michael L. (1979). Bidirectional synonym ratings of 464 noun pairs. *Journal of Verbal Learning and Verbal Behaviour*, 18:109–127.

Wierzbicka, Anna (1972). *Semantic Primitives*. Athenäum-Verlag, Frankfurt.

Wittgenstein, Ludwig (1953). *Philosophical Investigations*. Blackwell, Oxford.

Yarowsky, David (1992). Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING-92)*, pages 454–460.

Yarowsky, David (1994). Decision lists for lexical ambiguity resolution. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 88–95.

Zadeh, Lotfi (1965). Fuzzy sets. *Information and Control*, 8:338–353.

Zernik, Uri, editor (1991). *Lexical Acquisition: Exploiting On-line Resources to Build a Lexicon*. Lawrence Erlbaum Associates.

Appendix A

The ontology

```
(in-package :it)
(cc it)

;;;======================================================
;;; This is the ontology of the system
;;;======================================================

;;;------------------------------------------------------
;;; some fuzzy sets   (not part of Loom!)
;;;------------------------------------------------------

(deffuzzyset unknown :mu (lambda (x) 1.0))
(deffuzzyset undefined :mu (lambda (x) 0.0))

(deffuzzyset high :mu (lambda (x) (cond ((< x 0.4) 0.0)
                                        ((< x 0.9) (/ (- x 0.4) 0.5))
                                        (t 1.0))))

(deffuzzyset low :mu (lambda (x) (cond ((< x 0.1) 1.0)
                                       ((< x 0.6) (/ (- x 0.6) -0.5))
                                       (t 0.0))))

(deffuzzyset medium :mu (lambda (x) (cond ((< x 0.05) 0.0)
                                          ((< x 0.45) (/ (- x 0.05) 0.4))
                                          ((< x 0.55) 1.0)
                                          ((< x 0.95) (/ (- x 0.95) -0.4))
                                          (t 0.0))))

;;; (Would like to define a 'fuzzy concept', but can't in Loom)
(defconcept low-high-scale :is (:or (:through 0.0 1.0)))

;;;------------------------------------------------------
;;;   Top-level
;;;------------------------------------------------------

(defconcept MyThing)
(defconcept Object :is-primitive MyThing)
(defconcept Quality :is-primitive MyThing)
(defconcept Situation :is-primitive MyThing)
```

```
;;;--------------------------------------------------------
;;;    Object taxonomy
;;;--------------------------------------------------------

(defconcept Living-thing :is-primitive Object)
(defconcept Animate-being :is-primitive Living-thing)
(defconcept Inanimate-being :is-primitive Living-thing)
(defconcept Person :is-primitive Object)
(defconcept Abstraction :is-primitive Object)
(defconcept Control :is-primitive Abstraction)

(defconcept Relationship :is-primitive Abstraction)
(defrelation RELATUM1 :domain Relationship :range MyThing)
(defrelation RELATUM2 :domain Relationship :range MyThing)
(defconcept Companionship :is-primitive Relationship)

(defconcept Container :is-primitive Object)
(defrelation CONTAINS :domain Container :range Object)
(defconcept GeographicLocal :is-primitive Container)
(defconcept Tract-Of-Land :is-primitive GeographicLocal)
(defconcept Tract-Of-Trees :is (:and Tract-Of-Land
                             (:some CONTAINS Tree)))
(defconcept Tree :is-primitive Object)

(defconcept Alcoholic :is-primitive Person)

;;;--------------------------------------------------------
;;;    Quality taxonomy
;;;--------------------------------------------------------

(defconcept Polar :is-primitive (:and (:at-most 1 POLE)))
(defrelation POLE :domain Polar :range Poles
             :characteristic :single-valued)
(defconcept Poles :is (:one-of positive negative))

(defconcept Scalable :is-primitive (:at-most 1 DEGREE))
(defrelation DEGREE :domain Scalable :range ScaleValue
             :characteristics :single-valued)
(defconcept ScaleValue :is (:through 0.0 1.0))

;;; status is a quality of something that is on or off
(defconcept Status :is-primitive (:and Quality Polar))
;;; an attribute is an ascribed quality
(defconcept Attribute :is-primitive (:and Quality Scalable))
;;; a trait is a clearly-defined essential quality of a person
(defconcept Trait :is-primitive (:and Quality Scalable Polar))

(defrelation QUALITY :domain MyThing :range Quality)
(defrelation ATTRIBUTE :is-primitive QUALITY
             :domain MyThing :range Attribute)
(defrelation ATTRIBUTE-OF :is (:inverse ATTRIBUTE))
(defrelation HAS-TRAIT :is-primitive QUALITY
             :domain Person :range Trait)
```

```
(defconcept Power :is-primitive Attribute)
(defconcept Authority :is (:and Power
    ;;; can have at most 1 ATTRIBUTE
                               (:at-most 1 ATTRIBUTE)
                               (:all ATTRIBUTE (:or Peremptory Official))))

(defconcept Urgency :is-primitive Attribute)
(defconcept Endurance :is-primitive Attribute)
(defconcept Correctness :is-primitive Attribute)
(defconcept InCorrectness :is-primitive Attribute)
(defconcept Carefulness :is-primitive Attribute)

(defconcept Official :is-primitive Quality)
(defconcept Peremptory :is-primitive Quality)
(defconcept Arbitrary :is-primitive Peremptory)
(defconcept Imperative :is-primitive Attribute)

(defconcept Courage :is-primitive Trait)
(defconcept Boldness :is-primitive Trait)

(defconcept Deviation :is-primitive Attribute)
(defconcept Severity :is-primitive Attribute)
(defconcept Blameworthiness :is-primitive Quality)
(defconcept Stupidity :is-primitive Quality)
(defconcept VeryStupidity :is-primitive Stupidity)

(defconcept Mercy :is-primitive Quality)
(defconcept Desolate :is-primitive Quality)
(defconcept Courtesy :is-primitive Attribute)
(defconcept Equal :is-primitive Attribute)
(defconcept Subordinate :is-primitive Attribute)
(defconcept Nonconformity :is-primitive Attribute)
(defconcept Sinful :is-primitive Attribute)
(defconcept Significance :is-primitive Attribute)
(defconcept Categorical :is-primitive Attribute)
(defconcept Questionable :is-primitive Attribute) ;; opposite of categorical

(defconcept Size :is-primitive Attribute)
(defconcept Wildness :is-primitive Attribute)
(defconcept Proximity :is-primitive Attribute)

(defconcept Thin :is-primitive Attribute)

;;;-------------------------------------------------------
;;;    State taxonomy
;;;-------------------------------------------------------

(defconcept State
    :is-primitive Situation)

(defconcept Binary-state
    :is (:and State
              (:exactly 1 STATE-OBJECT)
              (:exactly 1 STATE-VALUE)))

(defrelation STATE-OBJECT :domain Binary-state :range Object)
(defrelation STATE-VALUE :domain Binary-state :range Quality)

;;;------------------------------------------------------
;;;    Activity taxonomy
;;;------------------------------------------------------

(defconcept Activity
    :is-primitive (:and Situation
                        (:at-most 1 ACTOR)
                        (:at-most 1 ACTEE)
                        (:at-most 1 ACT-CAUSER)))
(defrelation ACTOR :domain Activity :range MyThing)
(defrelation ACTEE :domain Activity :range MyThing)
(defrelation ACTOR-OF :is (:inverse ACTOR))
(defrelation ACTEE-OF :is (:inverse ACTEE))
(defrelation ACT-CAUSER :domain Activity :range Person)
(defrelation BENEFICIARY :domain Activity :range MyThing)
(defrelation CAUSE :domain Activity :range MyThing)
(defrelation CAUSE-OF :is (:inverse CAUSE))

(defconcept Communicate
    :is (:and Activity
              (:exactly 1 SAYER)
              (:exactly 1 SAYEE)
              (:exactly 1 SAYING)))
(defrelation SAYER :is (:and ACTOR (:range Person)))
(defrelation SAYEE :is (:and ACTEE (:range Person)))
(defrelation SAYING :domain Communicate :range MyThing)

(defconcept GroupActivity
  :is-primitive (:and Activity
                      (:at-most 1 CO-ACTOR)))
(defrelation CO-ACTOR :domain GroupActivity :range MyThing)

(defconcept Generic-Error :is (:and Activity
                        ;;  (:at-most 0 ACTEE)  ;; no actee (but doesn't work)
                            (:all ACTOR Person) ;; ACTOR is Person
                            (:some ATTRIBUTE Deviation)))

(defconcept Perform :is-primitive Activity)
(defconcept Make :is-primitive Activity)

(defconcept Criticism :is-primitive Activity)
(defconcept Misconception :is-primitive (:and Activity))
(defconcept Accident :is-primitive (:and Activity))
(defconcept Inattention :is-primitive (:and Activity))

(defconcept Quit :is-primitive Activity)
(defconcept Surrender :is-primitive Activity)
(defconcept Leave :is-primitive Activity)
(defconcept Occupy :is-primitive Activity)
```

```
(defconcept Renunciate :is-primitive Activity)

(defconcept Go-concept :is-primitive Activity)
(defconcept Go-With :is-primitive GroupActivity)
(defconcept Guide :is-primitive Activity)
(defconcept Protect :is-primitive Activity)
(defconcept Supervise :is-primitive Activity)

(defconcept Statement :is-primitive Communicate)
(defconcept Untruth :is (:and Statement
                              (:some ATTRIBUTE Nonconformity)))
(defconcept Contradict :is-primitive Activity)
(defconcept Intend :is-primitive Activity)
(defconcept Aware :is-primitive Activity)
(defconcept Mislead :is-primitive Activity)
(defconcept FaceSave :is-primitive Activity)
(defconcept Contradict :is-primitive Activity)

(defconcept Fight :is-primitive GroupActivity)
(defconcept Warn :is-primitive Activity)

;;;-------------------------------------------------------
;;;    Event taxonomy
;;;-------------------------------------------------------

(defconcept Event
    :is-primitive (:and Situation
                        (:exactly 1 EV-PRE-STATE)
                        (:exactly 1 EV-POST-STATE)
                        (:at-most 1 EV-ACTIVITY)))

(defrelation EV-PRE-STATE :domain Event :range State)
(defrelation EV-POST-STATE :domain Event :range State)
(defrelation EV-ACTIVITY :domain Event :range Activity)

(defconcept Transition
    :is (:and Event
              (:at-most 0 EV-ACTIVITY)))

;;;-------------------------------------------------------
;;; This contains the components of the ontology based on the merged
;;; ontology of CYC and Penman's Upper Model.
;;;-------------------------------------------------------

;;; Attributes
;;;---------------------
(defconcept Mental-Attribute :is-primitive Attribute)
(defconcept Feeling-Attribute :is-primitive Mental-Attribute)

;;; Could store with this the 'recipient' of the hostility, but I've put
;;; that in a Feels-Towards-Object
(defconcept Hostility :is-primitive Feeling-Attribute)
(defconcept Hatred :is-primitive Feeling-Attribute)
```

```
;;; Situations/Events/Activities
;;;-------------------------------------

;;; Feeling (emotions) is a relation in the merged ontology.
;;; (probably because feel, like 'is', is a copula verb.
(defconcept Feels-Emotion-Towards
  :is-primitive (:and Situation
                      (:exactly 1 SENSER)
                      (:exactly 1 EMOTION)
                      (:at-least 1 TOWARDS)))
(defrelation SENSER :domain Feels-Emotion-Towards :range Person)
(defrelation SENSER-OF :is (:inverse SENSER))
(defrelation EMOTION :domain Feels-Emotion-Towards :range Feeling-Attribute)
(defrelation TOWARDS :domain Feels-Emotion-Towards :range MyThing)

(defconcept Hate
  :is (:and Feels-Emotion-Towards
            (:some EMOTION Hatred)))

;;; Objects
;;;------------------------

(defconcept Enemy :is (:and Person
                            (:some SENSER-OF (:and Feels-Emotion-Towards
                                                   (:some EMOTION Hostility)))))

;;;------------------------------------------------------
;;; Domain concepts
;;;------------------------------------------------------

(defconcept John :is-primitive Person)
(defconcept Mary :is-primitive Person)
```

Appendix B

Cluster representations

```
(defcluster say_C
    :syns (tell_l say_l)
    :core (ROOT Communicate
                (SAYER (V1 Person))
                (SAYING (V2 Activity)))
    :covers (ROOT)
    )
```

```
(defcluster make_C
    :syns (make_l)
    :core (ROOT Make
                (ACTOR (V1 Person))
                (ACTEE (V2 MyThing)))
    :covers (ROOT)
    )
```

```
(defcluster accompany_C
    ;;;       from Gove
    :syns (accompany_l attend_l conduct_l escort_l convoy_l chaperon_l)
    :core (ROOT Go-With
                (ACTOR (V1 Person))
                (CO-ACTOR (V2 Person)))
    :covers (ROOT)
    :periph ( (P1 Companionship (RELATUM1 V1) (RELATUM2 V2))
              (P2 Status (ATTRIBUTE-OF V1))  ;; equal or subordinate
              (P3 Courtesy (ATTRIBUTE-OF ROOT))
              (P4 Guide (ACTOR V1) (ACTEE V2))
              (P5 Protect (ACTOR V1) (ACTEE V2))
              (P6 Supervise (ACTOR V1) (ACTEE V2))  ;; is motive of V1
              ;;; propriety as motive?
             )

    :distinctions
    (
     (accompany_l always medium implication P1)
     (accompany_l usually medium implication (P2 (ATTRIBUTE (X Equal))))
     (attend_l usually medium implication (P2 (ATTRIBUTE (X Subordinate))))
     (attend_l sometimes medium suggestion P3)
     (conduct_l usually medium implication P4)
     ;;; escort_l adds-to accompany_l (always medium implication P5)
     (escort_l usually medium implication (P2 (ATTRIBUTE (X Equal))))
     (escort_l usually medium suggestion P3)
     (escort_l always medium implication P5)
     ;;; convoy_l adds-to accompany_l (always medium implication P5)
     (convoy_l never medium suggestion P3)
     (convoy_l usually medium implication (P2 (ATTRIBUTE (X Equal))))
     (convoy_l always medium implication P5)
     (chaperon_l sometimes medium suggestion P6)
     )
    )
```

```
(defcluster alcoholic_C
    ;;;       from Gove and Hayakawa
    :syns (alcoholic_l drunkard_l drunk_l boozer_l inebriate_l
                       dipsomaniac_l sot_l lush_l soak_l)
    :core (ROOT Alcoholic)
    :covers (ROOT)

    :distinctions
    (
     (inebriate_l high formality)
     (dipsomaniac_l high formality)
     (drunkard_l medium formality)
     (alcoholic_l medium formality)
     (boozer_l low formality)
     (drunk_l low formality)
     (sot_l low formality)
     (lush_l low formality)

     (inebriate_l high floridity)
     (dipsomaniac_l high floridity)
     (drunk_l low floridity)

     ;; (concreteness from my intuition)
     (boozer_l high concreteness)
     (drunkard_l high concreteness)
     (alcoholic_l low concreteness)
     (dipsomaniac_l low concreteness)

     (drunk_l always strong pejorative ROOT)
     (drunkard_l always medium pejorative ROOT)
     (sot_l usually medium pejorative ROOT)
     (lush_l always weak pejorative ROOT)
     (alcoholic_l always medium neutral ROOT)
     (boozer_l always medium favourable ROOT)
     (soak_l always medium favourable ROOT)
     )
    )
```

```
(defcluster enemy_C
    ;;;       from Gove
    :syns (enemy_l foe_l)
    :core (ROOT Enemy)
    :covers (ROOT)
    :p-link ((V1 ;; ROOT is the enemy of V1
              (:and (Person V1)
                    (:for-some (?x ?y) (:and (Feels-Emotion-Towards ?x)
                                             (SENSER ?x ROOT)
                                             (TOWARDS ?x V1)
                                             (EMOTION ?x ?y)
                                             (Hostility ?y)))))
             (V2 ;; the hostile feeling
              (:and (Hostility V2)
                    (:for-some ?x (:and (Feels-Emotion-Towards ?x)
                                        (SENSER ?x ROOT)
                                        (EMOTION ?x V2)))))
             )
    :periph ((P1 Hate (SENSER ROOT) (TOWARDS V1) (CAUSE-OF V2))
             (P2 Fight (ACTOR ROOT) (CO-ACTOR V1))
             )

    :distinctions
    (
     (enemy_l usually medium implication P1)
     (foe_l sometimes medium implication P1)

     (enemy_l never medium implication P2)
     (foe_l always medium implication P2)

     (foe_l high formality)
     )
    )
```

```
(defcluster error_C
    ;;;       from Gove
    :syns (error_l mistake_l blunder_l slip_l lapse_l howler_l)
    ;;:syns (blunder_l)
    :core (ROOT Generic-Error)
    :covers (ROOT)
    :p-link ((V1 (:and (Person V1) (ACTOR ROOT V1)))
             (V2 (:and (Deviation V2) (ATTRIBUTE ROOT V2))))
    :periph ((P1 Stupidity (ATTRIBUTE-OF V1))
             (P2 Blameworthiness (ATTRIBUTE-OF V1))
             (P3 Criticism (ACTEE V1) (ATTRIBUTE (P3-1 Severity)))
             (P4 Misconception (CAUSE-OF V2) (ACTOR V1))
             (P5 Accident (CAUSE-OF V2) (ACTOR V1))
             (P6 Inattention (CAUSE-OF V2) (ACTOR V1))
             )

    :distinctions
    (
     ;;  Blunder commonly implies stupidity.
     (blunder_l usually medium implication P1)
     ;;  Mistake does not always imply blameworthiness, blunder sometimes
     (mistake_l sometimes medium implication (P2 (DEGREE 'medium)))
     (blunder_l sometimes medium implication (P2 (DEGREE 'high)))

     ;;  Mistake implies less severe criticism than error.
     ;;  Blunder is harsher than mistake or error.
     (mistake_l always medium implication (P3-1 (DEGREE 'low)))
     (error_l always medium implication (P3-1 (DEGREE 'medium)))
     (blunder_l always medium implication (P3-1 (DEGREE 'high)))

     ;;  Mistake implies misconception.
     (mistake_l always medium implication P4)

     ;;  Slip carries a stronger implication of accident than mistake.
     ;;  Lapse implies inattention more than accident.
     (slip_l always medium implication P5)
     (mistake_l always weak implication P5)
     (lapse_l always weak implication P5)
     (lapse_l always medium implication P6)

     ;;  Blunder expresses a pejorative attitude towards the person.
     (blunder_l always medium pejorative V1)

     ;;  Blunder is a concrete word, error and mistake are abstract.
     (blunder_l high concreteness)
     (error_l low concreteness)
     (mistake_l low concreteness)

     ;;  Howler is an informal term.
     (howler_l low formality)
     )
    )
```

```
(defcluster err_C
    ;;;       from my intuition (cf. Gove on error)
    :syns (err_l blunder-v_l)
    :core (ROOT Make
                (ACTOR (V1 Person))
                (ACTEE (V2 Generic-Error)))
    :covers (ROOT V2)
    :periph ((P1 Stupidity (ATTRIBUTE-OF V1)))

    :distinctions
    (
     (blunder-v_l usually medium implication P1)
     (blunder-v_l always medium pejorative V1)
     (err_l high formality)
     ))
```

```
(defcluster tract_of_trees_C
    ;;;       from Room and Dimarco+Hirst+Stede
    ;;;       includes German synonyms
    :syns (forest_l woods_l Wald_l Geholz_l)
    :core (ROOT Tract-Of-Trees)
    :covers (ROOT)
    :periph ((P1 Size (ATTRIBUTE-OF ROOT))
             (P2 Wildness (ATTRIBUTE-OF ROOT))
             (P3 Proximity (ATTRIBUTE-OF ROOT))
             ;;; undergrowth, type of trees
             )
    :distinctions
    (
     (forest_l always medium denotation (P1 (DEGREE 'high)))
     (forest_l always medium denotation (P2 (DEGREE 'high)))
     (forest_l always medium denotation (P3 (DEGREE 'low)))

     (woods_l always medium denotation (P1 (DEGREE '(OR low medium))))
     (woods_l always medium denotation (P2 (DEGREE 'low)))
     (woods_l always medium denotation (P3 (DEGREE '(OR medium high))))

     (Wald_l always medium denotation (P1 (DEGREE '(OR medium high))))
     (Wald_l always medium denotation (P2 (DEGREE '(OR medium high))))
     (Wald_l always medium denotation (P3 (DEGREE '(OR low medium))))

     (Geholz_l always medium denotation (P1 (DEGREE 'low)))
     (Geholz_l always medium denotation (P2 (DEGREE 'low)))
     (Geholz_l always medium denotation (P3 (DEGREE 'high)))
     )
    )
```

```
(defcluster order_C
    ;;;       from Gove
    :syns (command_l order_l bid_l enjoin_l direct_l instruct_l)
    :core (ROOT Communicate
                (SAYER (V1 Person))
                (SAYEE (V2 Person))
                (SAYING (V3 Perform
                            (ACTOR V2)
                            (ACTEE (V4 Activity (ACTOR V2))))))

    :covers (ROOT V3)

              ;; can be either peremptory or official authority (not both)
    :periph ( (P1 Authority (ATTRIBUTE-OF V1))
              (P2 Imperative (ATTRIBUTE-OF V4))
              (P3 Warn (ACTOR V1) (ACTEE V2))
              )

    :distinctions
    (
     (command_l always medium implication (P1 (ATTRIBUTE (X Official))))
     (order_l always medium implication (P1 (ATTRIBUTE (X Peremptory))))
     (bid_l usually medium suggestion (P1 (ATTRIBUTE (X Peremptory))))

     (command_l always medium denotation (P2 (DEGREE 'high)))
     (order_l always medium denotation (P2 (DEGREE 'high)))
     (enjoin_l always medium denotation (P2 (DEGREE 'medium)))
     (direct_l always medium denotation (P2 (DEGREE 'medium)))
     (instruct_l always medium denotation (P2 (DEGREE 'medium)))

     (enjoin_l always medium implication P1)
     (enjoin_l always medium implication P3)

     (instruct_l high formality)
     (enjoin_l high formality)
     (direct_l medium formality)

     (instruct_l high concreteness)
     (direct_l high concreteness)
     (order_l medium concreteness)
     (command_l medium concreteness)
     )
    )
```

```
(defcluster thin_C
    ;;;       from Gove and Hayakawa, but glosses over many distinctions
    :syns (skinny_l thin_l slim_l slender_l)
    :core (ROOT Thin)
    :covers (ROOT)
    :p-link ((V1 (:and (Person V1) (ATTRIBUTE-OF ROOT V1))))
    :periph ()

    :distinctions
    (
     (skinny_l usually medium pejorative V1)
     (thin_l usually medium neutral V1)
     (slim_l usually medium favourable V1)
     (slender_l usually strong favourable V1)
     )
    )
```

```
(defcluster untruth_C
    ;;;       from Gove and Hayakawa
    :syns (lie_l falsehood_l untruth_l misrepresentation_l prevarication_l fib_l)
    :core (ROOT Untruth)
    :covers (ROOT)
    :p-link ((V1 (:and (Person V1) (SAYER ROOT V1)))
             (V2 (:and (Nonconformity V2) (ATTRIBUTE ROOT V2)))) ;; to truth

    :periph ((P1 Contradict (ACTOR ROOT)) ;; contradiction of truth
             (P2 Intend (ACTOR V1)  ;; intends to mislead
                        (ACTEE (P2-1 Mislead)))
             (P3 Aware (ACTOR V1) (ACTEE V2))
             (P4 Criticism (ACTEE V1)
                           (ATTRIBUTE (P4-1 Severity)))
             (P5 FaceSave (ACTOR V1))
             (P6 Significance (ATTRIBUTE-OF ROOT))
             (P7 Misconception (ACTOR V1) (CAUSE-OF ROOT))
    )
    :distinctions
    (
     (lie_l always medium implication (P1 (ATTRIBUTE (X Categorical))))
     (lie_l always medium implication P2)
     (lie_l always medium implication P3)
     (lie_l always medium implication (P4-1 (DEGREE 'high)))
     (lie_l usually medium pejorative V1)

     ;; falsehood is less censorious (critical) than lie
     (falsehood_l always medium implication P3)
     (falsehood_l usually medium implication (P4-1 (DEGREE 'medium)))

     ;; untruth can imply same as lie OR it can imply a misconception
     (untruth_l sometimes medium implication P7)
     (untruth_l seldom medium implication P2)

     ;; fib is a trivial lie
     (fib_l sometimes medium implication P2)
     (fib_l always medium implication P3)
     (fib_l always medium implication (P4-1 (DEGREE 'low)))
     (fib_l usually medium implication P5)
     (fib_l always medium implication (P6 (DEGREE 'low)))
     (fib_l low formality)

     ;; misrepresentation is questionable (not categorical)
     (misrepresentation_l usually medium implication (P1 (ATTRIBUTE (X Questionable))))
     (misrepresentation_l usually medium implication P2)
     (misrepresentation_l always medium implication P3)

     ;; prevarication is a formal term, similar to misrepresentation
     (prevarication_l always medium implication (P1 (ATTRIBUTE (X Questionable))))
     (prevarication_l usually medium implication P2)
     (prevarication_l always medium implication P3)
     (prevarication_l high formality)

     (lie_l high concreteness)
     (falsehood_l low concreteness)
     (untruth_l low concreteness)
     )
    )
```

```
(defcluster tell-a-lie_C
    ;;;       from Gove
    :syns (lie-v_l fib-v_l prevaricate_l equivocate_l)
    :core (ROOT Communicate
                (SAYER (V1 Person))
                (SAYING (V2 Untruth)))
    :covers (ROOT V2)
    :p-link ((V1 (:and (SAYER V2 V1)))
             (V3 (:and (Nonconformity V3) (ATTRIBUTE V2 V3))))

    :periph ((P1 Contradict (ACTOR V2)) ;; contradiction of truth
             (P2 Intend (ACTOR V1)  ;; intends to mislead
                        (ACTEE (P2-1 Mislead)))
```

```
         (P3 Aware (ACTOR V1) (ACTEE V3))
         (P4 Criticism (ACTEE V1)
                       (ATTRIBUTE (P4-1 Severity)))
         (P5 Significance (ATTRIBUTE-OF V2))
)

:distinctions
(
 (lie-v_l always medium implication (P1 (ATTRIBUTE (X Categorical))))
 (lie-v_l always medium implication P2)
 (lie-v_l always medium implication P3)
 (lie-v_l always medium implication (P4-1 (DEGREE 'high)))
 (lie-v_l usually medium pejorative V1)

 (prevaricate_l usually medium implication (P1 (ATTRIBUTE (X Questionable))))
 (prevaricate_l sometimes medium implication P2)
 (prevaricate_l sometimes medium implication P3)
 (prevaricate_l always medium implication (P4-1 (DEGREE 'medium)))
 (prevaricate_l usually weak pejorative V1)

 (equivocate_l always medium implication (P1 (ATTRIBUTE (X Questionable))))
 (equivocate_l always medium implication P2)
 (equivocate_l always medium implication P3)

 (fib-v_l always medium implication (P5 (DEGREE 'low)))
 (fib-v_l always medium implication P2)
 (fib-v_l always medium implication P3)
 (fib-v_l always medium implication (P4-1 (DEGREE 'low)))

 (prevaricate_l low concreteness)
 (prevaricate_l high formality)
 (fib-v_l low formality)
 ))
```