LEXICAL CHOICE IN NATURAL LANGUAGE GENERATION

Jacques Robin

7 September 1990

Abstract

In this paper we survey the issue of lexical choice in natural language generation. We first define lexical choice as the choice of *open-class* lexical items (whether phrasal patterns or individual words) appropriate to express the content units of the utterance being generated in a given situation of enunciation. We then distinguish between five major classes of constraints on lexical choice: grammatical, inter-lexical, encyclopedic, interpersonal and discursive. For each class we review how they have been represented and used in existing generators. We conclude by pointing out the common limitations of these generators with respect to lexical choice as well as some directions for future research in the field.

1. Introduction

1.1 Natural Language Generation: a problem of choice

One day at about midday in the Parc Monceau district, on the back platform of a more or less full S bus (now #84), I observed a person with a very long neck who was wearing a felt hat which had a plaited cord around it instead of a ribbon. This individual suddenly addressed the man standing next to him, accusing him of purposely treading on his toes every time any passenger got on or off. However, he quickly abandoned the dispute and threw himself on to a seat which had become vacant.
Two hours later I saw him in front of the gare Saint-Lazare engaged in earnest conversation with a friend who was advising him to reduce the space between the lapels of his overcoat by getting a competent tailor to raise the top button.

In a bus of the S-line, 10m long, 3 wide, 6 high, at 3.6km from its starting point, loaded with 48 people, at 12:17pm, a person of the masculine sex aged 27 years 3 months and 8 days, 1,72m tall and weighing 65kg and wearing a hat 35cm in height round the crown of which was a ribbon 60cm long, interpellated a man aged 48 years 4 months and 3 days, 1,68m tall and weighing 77kg, by means of 14 words whose enunciation lasted 5 seconds and which alluded to some involuntary displacements of from 15 to 20mm. Then he went and sat down about 110cm away.
57mn later he was 10m away from the suburban entrance to the gare Saint-Lazare and was walking up and down over a distance of 30m with a friend aged 28, 1,70m tall and weighing 71kg who advised him in 15 words to move by 5cm in the direction of the zenith a button which was 3cm in diameter.

I got into a bus full of taxpayers who were giving some money to a taxpayer who had on his taxpayer's stomach a little box which allowed the other taxpayers to continue their taxpayers' journey. I noticed in this bus a taxpayer with a long taxpayer's neck and whose taxpayer's head bore a taxpayer's felt hat encircled by a plait the like of which no taxpayer ever wore before. Suddenly the said taxpayer, peremptorily addressed a nearby taxpayer complaining bitterly that he was purposely treading on his taxpayer's toes every time other taxpayers got on or off the taxpayers' bus. Then the angry taxpayer went and sat down in a seat for taxpayers which another taxpayer had just vacated.
Some taxpayer's hours later I caught sight of him in the Cour of a taxpayer de Rome, in the company of a taxpayer who was giving him some advice on the elegance of the taxpayer.

I was plat-bus-forming co-massitudinarily in a lutetio-meridional space-time and I was neighbouring a longisthmusical plaitroundthehatted greenhorn. Who said to a mediocranon: "you're jostleseeming me." Having ejaculated this he freeplaced himself voraciously.
In a posterior spatio-temporality I saw him again with an X who was saying: "You ought to buttonsupplement your overcoat." And he whyexplained him.

Figure 1-1: Excerpts from Raymond Queneau's *Exercises in style*

The four texts of figure 1-1, four different versions of the same little narrative, are due to Raymond Queneau. In his "Exercises in Style" [Queneau 58], he gives 99 different versions of that simple story. In addition to being hilarious from cover to cover, this book strikingly illustrates the incredible number of different ways to express the same content in natural language. Given the tremendous expressive diversity natural language provides, **Natural Language Generation (NLG)** meets a twofold challenge: the challenge of representing this diversity and the challenge of controlling it. This twofold challenge is best defined as a problem of choice: it requires identifying all the possible ways to express a given content and *choosing* among them the most appropriate in a particular context.

In the literature (e.g. in [McKeown 85a], [McDonald, Vaughan & Pustejovski 87], [Mann 87], [Watt 88]), text generation[1] has often been decomposed into three subtasks:

- **Content determination**: answering the question "what to say ?", content determination involves choosing what entities of the underlying application to talk about in the text. It also involves choosing which properties of these entities to express in the text.
- **Discourse organization**: answering the question "when to say what ?", discourse organization involves choosing how to group content units together, choosing in what order to present them in the discourse etc. Discourse organization is specific to text generation
- **Surface realization**: answering the question "how to say it ?", surface realization involves choosing how to map content units onto **Surface Linguistic Units (SLUs)**. It involves choosing the type of these linguistic units (i.e. sentences, clauses, nominals[2], **Prepositional Phrases (PPs)** etc), their grammatical form (e.g. active vs. passive for a clause, definite vs. indefinite for an NP) and the lexical items they are made of.

In order to illustrate this categorization of generation choices, let us determine the similarities and differences of our four example texts, with respect to each type of generation choice:

- *In terms of content determination*, as four different versions of the same story, all four texts describe the same event. However, the second text convey a much greater amount of information about each aspect and participant of the story than the three other ones. Indeed, the humoristic effect of this second text precisely comes from the overly excessive amount of details it contains.
- *In terms of discourse organization*, all four texts follow a similar chronological structure.
- *In terms of surface realization*, the grammatical forms of the linguistic units they are made of does not differ much (past tense first person narrative for all four texts). However, these four different texts result from four drastically different sets of word choices. The humorous effect of the third and fourth texts comes from two extreme word choice strategies. In the third one the use of a single underspecific word to refer to every human participant and in the fourth one the pervasive use of compound neologisms.

Surface realization in turn can be decomposed into four subtasks:

- **Lexicalization**: the expression of semantic content by open-class[3] lexical items.
- **Semantic grammaticalization**: the expression of semantic content by grammatical devices.
- **Morpho-syntactic grammaticalization**: the enforcement of syntax, often involving

[1] i.e. the generation of paragraph-length texts as opposed to single sentence generation.

[2] i.e. **Noun Phrases (NPs)** and pronouns.

[3] cf. infra for the definition of open-class items.

morphologicalization.

- **Linearization**: the actual spelling of sentences as strings of morphologicalized lexical items.

Consider for example the generation of the utterance *Who is this giant ?* expressing a request concerning the identity of a human male of extreme height. For this utterance:

- Lexicalization involves using the noun *giant*[4] to express the fact that the object of the request is a human male of extreme height and of the verb *to be* to express that the request is about its identity.
- Semantic grammaticalization involves using the interrogative mood to signify that the utterance is a request and using the third singular person to express that the object of the request is a single individual which is neither the speaker nor the hearer.
- Morpho-syntactic grammaticalization involves enforcing verb-subject agreement by using the inflection *is* of the verb *to be*.
- Linearization involves following the ordering pattern English syntax dictates for questions, namely WH-SUBJECT.VERB.OBJECT.

As illustrated by the examples we have presented, each generation subtask (content determination, discourse organization, lexicalization, grammaticalization and linearization) involves performing some type of choice. This paper is a survey of the choices involved in the lexicalization subtask. In the literature they are generally called **lexical choices**, or alternatively *word choices*, *lexical selections* or *lexical decisions*.

1.2 What is lexical choice ?

Giving a more precise definition of lexical choice requires making two distinctions: the distinction between **open-class words** and **closed-class words** and the distinction between **microcoded generators** and **macrocoded generators**.

From a grammatical perspective the most fundamental way to categorize words is by **Part-Of-Speech (POS)**. POS like conjunctions and determiners are closed classes of words. They form a finite system of a few elements whose function is to mark the grammatical form of the SLUs they appear in. The application of grammar rules *entirely dictate* choice of most closed-class word, with a few exceptions[5]. On the other hand, POS like verbs, nouns, adjectives and adverbs are open classes of words. They form a potentially infinite system of elements whose function is independent of the grammatical form of the SLUs they appear in. Thus, the application of grammar rules cannot *entirely dictate* choice of open-class word, but *only restrict* it. In this paper, we limit the definition of lexical choice to choices not completely dictated by grammatical choice[6]. Therefore we only consider choices of open-class words as lexical, choice of closed-class word being purely grammatical. However, as an exception to this rule, we consider two phenonema, pronominalization and ellipsis, resulting in choices of closed-class items as cases of lexical choices. **Pronominalization** refers to the use of **central pronouns**[7] in place of either NP head nouns or entire NPs. Central pronouns form a closed class of words and once the decision of using a central pronoun has been

---

[4]as opposed to, say, the phrase *very tall guy*.

[5]see below.

[6]i.e. the choices performed to carry out grammaticalization.

[7]i.e. personal, reflexive and possessive pronouns.

made, choosing the appropriate one is entirely dictated by grammatical constraints. However, the initial choice of a central pronoun over an NP is not. As a choice between a closed-class item (the pronoun) and an open-class item (the NP), pronominalization is dictated by various non-grammatical factors (including the cohesive constraints we survey in section 7.2). We thus also consider it a type of lexical choice. Similarly, it is also convenient to consider some cases of ellipsis as lexical choice. **Ellipsis** refers to the omission of a constituent in a clause. Although strigently restricted by grammatical constraints, for NLG purposes, ellipsis is best viewed as the choice of the empty string to implicitly convey the content unit that would have been explicitly conveyed by the ellided constituent. It is thus also a case of alternative between an open-class item (the ellided constituent) and a closed-class one (the empty string forming a singleton class).

Up to now we have talked of lexical choice only as choice of individual words. However, most complete text generators[8] are macrocoded. In macrocoded generators, atomic content units are realized by phrasal patterns, and thus choice of individual words does not occur. The lowest level linguistic units fully constructed by software at generation time are sentences, every lower level units (clauses, nominals, PPs etc) being constructed by hand at knowledge acquisition time and frozen into a phrasal lexicon. At generation time these units are retrieved as a whole from this phrasal lexicon. To the contrary, in microcoded generators, linguistic units of all levels are fully constructed at generation time from a word-based lexicon. Between fully instantiated phrasal patterns and individual words, there is a continuum of possibilities; the more variables there are in the phrasal patterns of a generator's lexicon, the more it is microcoded. Moreover, in some generators such as TEXT [McKeown 85b] and ROMPER [McCoy 86], some categories of SLUs are macrocoded in the lexicon, while others are microcoded during generation. A choice of phrasal pattern is a lexical choice just as a choice of individual word is.

We can now redefine lexical choice as "choice of SLUs from the following categories: clause patterns, nominal patterns, PP patterns, adjectival phrases patterns at the macrocoding level; verbs, nouns, central pronouns, adjectives, and adverbs at the microcoding level".

1.3 Organization of the survey

The lexical choice abilities of a generator can be evaluated along three main dimensions:

- The types of content units the generator can express.
- The categories of SLUs the generator is able to choose from.
- The types of constraints the generator can take into account to perform these choices.

A survey of the lexical choice abilities of existing generators can be organized along any of these three dimensions. Many different grammatical frameworks have been proposed in the literature. Besides POS, not many SLU categorization criteria are acknowledged in every one of these frameworks. Thus, any substantially refined categorization of SLUs requires a commitment to some grammatical framework. Just as SLU categorization is bound to be framework-specific, content units classification is bound to be domain-specific. In order to remain as neutral as possible with regard to both grammatical frameworks or application domains, the present survey is organized around the notion of constraint. Such an organization

[8]By complete text generators we mean generation systems performing content determination *and* text organization *and* surface realization (e.g. TEXT [McKeown 85b], PAULINE [Hovy 87], or ANA [Kukich 83]) as opposed to stand-alone generation components dedicated to only one or two of these subtasks (e.g. NIGEL [Mann & Matthiessen 83], FUF [Elhadad 88] or MUMBLE [McDonald 80]).

is also convenient because NLG is best viewed *as a process of making choices under constraints*[9]. There are two main reasons for that:

- Choices of different types[10] are largely intertwined; performing one type of choice *constrains* the alternatives for the other types of choice (e.g. choosing the passive voice for a sentence, a grammatical choice, constrain the choice of the main verb of that sentence, a lexical choice, to be a verb that can be passivized).

- External factors also constrain the alternatives for some types of choice (e.g. a user-model can be used to constrain lexical choices by specifying the words the user knows).

In the next section, we propose a classification of constraints on lexical choice in NLG. Each subsequent section covers a given constraint class. It identifies the constraints of that class and then explains how some of them have been used in existing generators. The conclusion of the paper includes a brief discussion on the adequacy of various generator architectures to implement the interaction between these different types of constraints.

[9]as pointed out by [McDonald 83] among others

[10]i.e. content determination choices, textual organizations choices, lexical choices, grammatical choices etc.

2. Classifying constraints on lexical choice

In this section, we propose a classification of the various constraints on lexical choice. This classification is based on three major distinctions. The first is between linguistic and situational constraints, the second between grammar and lexicon and the third between three aspects of the situation of enunciation.

2.1 Linguistic constraints

Every generator performing some form of lexical choice needs two sources of linguistic knowledge:

1. A grammar embodying the syntax of a given natural language and allowing the generator to produce grammatical utterances,
2. A lexicon associating each concept of the underlying application domain with the various words that can be used to refer to them in a given natural language.

These two sources of knowledge constitutes the first two sources of constraints on lexical choice. These constraints are *linguistic*, in the sense that they depend only upon the characteristics of the natural language used by the generator.

Grammar constrains lexical choice because many words have idiosyncratic grammatical properties. These properties restrict the grammatical forms of the utterance that can be generated using these words. Thus, once a generator has chosen the grammatical form of an utterance, the choice of a lexical item expressing a given content unit is *constrained* to those items that can appear in such a grammatical form. For example once a generator has committed itself to the use of a passive form for a sentence, the choice of that sentence's main verb is *constrained* to those verbs that can be passivized. Thus a grammatical choice can constrain a lexical choice and vice-versa.

Similarly, a lexical choice can constrain another lexical choice. This is because words also have idiosyncratic *lexical* properties. That is, independently of any grammatical or semantic considerations, many word pairs preferably appear next to one another in the well-formed phrases of a given natural language. Consequently, when one element of such pair is chosen by the generator to express some content unit, the word expressing some other content unit of the utterance to generate is *constrained* to be the other element of that pair. For example a generator of English that has committed itself to the use of the noun *dream* to express the indirect object of a dreaming action is constrained to choose the verb *to have* to express the dreaming action itself, because in correct English one says *to <u>have</u> a dream*, as opposed to, *to <u>do</u> a dream*[11] or to *to <u>dream</u> a dream*[12].

There are therefore two main types of linguistic constraints on lexical choice: **grammatical constraints** due to dependencies between grammatical and lexical choices and **inter-lexical constraints** due to dependencies among lexical choices.

---

[11]like in a literal translation of French.

[12]like in a literal translation of Hebrew.

2.2 Situational constraints

One of the most immediate observations about natural language utterances[13] is how much their meaning is dependent upon the situation in which they are enunciated. Consider the following utterance:
(1) *Sirocco rode the Titanide for one hour*.
Depending upon the encyclopedic knowledge shared by the speaker and the hearer of that utterance, its sense can be quite different. For example, if they share the knowledge that *Titanide* is a motorcycle manufacturer the sense of *to rode* and of *one hour* is different than if they share the knowledge that *Titanide* is a species of intelligent centaur-like hermaphrodite animals inhabiting Titan, the largest satellite of the planet Saturn. Similarly, if they share the knowledge that Sirocco dislikes Titanide riding, the sense of the utterance is different than if they share the knowledge than she loves it. The apparent paradox that the *same* utterance has *different* senses led many semanticists ([Lyons 77], [Barwise & Perry 83], [Ducrot 84] and [Raccah 87] among others) to propose a distinction between:

- **Utterance-type**: a grammatical string of words of a given natural language, considered independently of any particular situation of enunciation.
- **Utterance-token**: a grammatical string of words of a given natural language, as enunciated in a particular situation.

This distinction allows different utterance-tokens instances of the *same* utterance-type to have *different* senses. Since the sense of an utterance-token does not only depend on the meaning of the utterance-type it is an instance of, but also on the particular situation in which that token is enunciated, a generator aiming to produce an utterance (token) in a precise sense, must have some model of the situation of enunciation. Without such a model, the generator would not be able to guarantee that the chosen words and grammatical form for the generated utterance convey the desired sense *in that particular situation*. Hence, besides the grammar and the lexicon, a third source of constraints is needed for a generator to adequately perform lexical choice: a model of the situation of enunciation.

Utterance (1) above illustrated that the encyclopedic knowledge shared by the speaker and the hearer is one aspect of the situation of enunciation with influence on the sense of an utterance. Utterance (2) and (3) below illustrates that there are several other aspects of that situation with such an influence:
(2) *You will go to Babylon tomorrow*.
(3) *Being able to deal with the above examples raises the following issues*.
The sense of (2) depends on the social and/or emotional relationship between the speaker and the hearer. The knowledge of that relationship is necessary to interpret whether (2) is a promise, a treat, an order, a request, an assertion, etc. The sense of (3) depends on yet another aspect of the situation: the discourse surrounding (3). The knowledge of the sentences preceding and following (3) is necessary to identify what *examples* and *issues* are referred to in (3). As these three example utterances suggest, there are many different aspects in the situation of enunciation of an utterance that need to be taken into account to interpret or generate it. There have been numerous studies of the situation of enunciation in the language sciences literature. In his survey of that literature, [Elhadad 87] proposes a decomposition of the situation of enunciation in three major aspects. Independently, investigating the type of model of the situation of enunciation needed to constrain grammatical choice in unrestricted[14] NLG, [Matthiessen 87] uses a similar

---

[13]We are using the term "utterance" to refer to SLUs of any level (word, phrase, sentence, paragraph). Also we do not assume any restriction on the type of natural language communication an utterance is part of: it can be either written or oral, and involve two or more participants. In what follows, we are therefore using the terms "the speaker" and "the hearer" in a generic sense, where "the speaker" can possibly denote a writer and the hearer can possibly denote one or several hearers or readers.

[14]Unrestricted in terms of domain of application.

threefold decomposition based on the grammatical meta-functions put forward by [Halliday 85]. Drawing upon these works, we also propose distinguishing three situations of enunciation: the **Encyclopedic Situation of Enunciation (ESE)**, the **Interpersonal Situation of Enunciation (ISE)** and the **Discursive Situation of Enunciation (DSE)**.

The **ESE** of an utterance can be defined as the state of the world in which the utterance is enunciated, with "the world" being either the actual physical world or its representation in the speaker and/or hearer's mind. A description of the ESE answers the questions: what are the entities of the underlying domain ? how are they related ? in what state are they currently in ? what knowledge of these entities does the hearer has ? It is knowledge of the ESE that allows the disambiguation of utterances like:
Sirocco rode the Titanide for one hour.
Since [Schank & Abelson 77][15] and [Charniak 77] among others, emphasized the need to provide NLP programs with encyclopedic knowledge for these programs to demonstrate any natural language competence, almost every NLP program incorporates some model of the ESE. This model is generally encoded using either a frame-like knowledge representation formalism (e.g. frames [Minsky 75], [Charniak 77], [Bobrow & Winograd 77], MOPs [Schank 82], semantic networks [Brachman 79]) or a possible-worlds logic (e.g. [Moore 80]). The limitations of these models are thus inherently linked to limitations of the state-of-the-art in world-knowledge representation.

The **ISE** of an utterance can be defined as the type of communication the utterance is used for. This type of communication depends on various factors including the communicative goals of the speaker and the hearer, their social relationship, etc. A description of the ISE answers the question: who am I talking to and with what purpose ? It is knowledge of the ISE that allows the disambiguation of utterances like:
You will go to Babylon tomorrow.
Most NLP programs incorporating a model of the ISE (e.g. [Allen & Perrault 80], [Appelt 85]) are based on Searle's speech-act theory (see [Searle 69] and [Searle & Vanderveken 85]) and on the **Artificial Intelligence (AI)** formalism of planning (see for example [Fikes & Nilsson 71] and [Sacerdoti 77]). However, other programs, not directly concerned with the speech-act aspect of the ISE have used special-purpose models (e.g. [Hovy 87]).

The **DSE** of an utterance can be defined as the textual or conversational surrounding of the utterance. From a generation perspective, a description of the DSE answers the questions: what has been said before ? what is planned to be said next ? how are both related ? It is knowledge of the DSE that allows the disambiguation of utterances like:
Being able to deal with the above examples raises the following issues.
Most models of the DSE incorporated in NLP programs attempt to represent the structure of the on-going discourse (e.g. [Grosz & Sidner 86], [Grosz, Joshi & Weinstein 81], [Reichman 85]) that the utterance is part of. In text generation, there is strong relation between the DSE model and the textual organization scheme.

[15] calling the ESE the scriptal situation.

2.3 Five types of constraints on lexical choice

In section 1 of the present chapter we saw that they are two linguistic knowledge sources providing constraints on lexical choice: the grammar and the lexicon itself. In section 2 we saw a non-linguistic knowledge source providing constraints on lexical choice: the model of the situation of enunciation. We also saw that this model can be divided in three main components, each corresponding to a different aspect of the situation: the ESE model, the ISE model and the DSE model. We have thus identified five major types of constraints on lexical choice, one for each knowledge source providing it:

- **Grammatical constraints**: constraints coming from the grammatical chooser. For example, once the grammatical chooser has chosen to a passive form for a sentence, the lexical chooser is constrained to pick that sentence's main verb among those that can be passivized.
- **Interlexical constraints**: constraints recursively coming from the lexical chooser. For example, once the lexical chooser has committed itself to the use of the noun *dream* to express the indirect object of a dreaming action, it is constrained to choose the verb *to have* to express the dreaming action itself.
- **Encyclopedic constraints**: constraints coming from the model of the ESE. For example an ESE model indicating which properties of a domain entity distinguishes it from any other domain entity, can constrain the generator to refer to the entity by mean of such a property (e.g. if the ESE model indicates that there is a single member of the orca specie in the domain, that member can referred to by mean of the simple definite NP *the orca*, whereas such reference would be ambiguous if the ESE model indicated that there are many orcas in the domain).
- **Interpersonal constraints**: constraints coming from the model of the ISE. For example, an ISE model indicating that one active communicative goal is to be sympathetic toward a hearer known to be pro-Palestinian, can constrain references to Israel to be worded: *the zionist invader*
- **Discursive constraints**: constraints coming from the model of the DSE. For example an DSE model indicating that the domain entity in focus in the current sentence have already been referred in the previous sentence, can constrain reference to that entity to be by mean of a pronoun, (e.g. as in *<u>The</u> <u>rookie</u> <u>Yugoslavian</u> <u>center</u> <u>Vlade</u> <u>Divac</u> scored 22 points. <u>He</u> also had 8 rebounds and 3 blocks*).]

Figure 2-1 shows the flow of constraints in a surface generator where all these five types of constraint would be available. In this figure, arrows are bi-directional because not only the content of the knowledge source constrains lexical choice, but conversely lexical choice constrain the content of the knowledge sources. For example, an assertion by the speaker that a given object has a given property, can trigger an update in the ESE model, namely the addition of this property to the hearer's set of beliefs about this object. Figure 2-1 provides a *conceptual* decomposition of surface generation into two subtasks (lexical choice and grammatical choice) and of the knowledge needed to carry them out into five sources (grammar, lexicon, ESE model, ISE model and DSE model). This conceptual decomposition will serve two purposes in the remaining of this survey:

- Clarify the attempts to compare existing generators which are often based on radically different architectures, described in inconsistent terminologies and carry out quite different generation tasks.
- Structure the survey in five subsequent chapters, each one examining in detail a given type of constraints on lexical choice.

```
                                |-------|
                                |       |
                                |       |
~~~~~~~~~~~~~                   V       V
(           )                |-------------|
( ESE Model )<-------------->|             |     ~~~~~~~~~~~
(           )                |   Lexical   |     (         )
                             |             |
~~~~~~~~~~~~~   |----------->|             |<----( Lexicon )
                |            |   Chooser   |     (         )
~~~~~~~~~~~~~   |  |-------->|             |     ~~~~~~~~~~~
(           )   |  |         |-------------|
( ISE Model )<--|  |                ^
(           )      |                |
~~~~~~~~~~~~~      |                |
                   |                |
~~~~~~~~~~~~~      |                V
(           )      |         |-------------|     ~~~~~~~~~~~
( DSE Model )<-----|         | Grammatical |     (         )
(           )                |             |<----( Grammar )
~~~~~~~~~~~~~                |   Chooser   |     (         )
                             |-------------|     ~~~~~~~~~~~
```

Figure 2-1: Five sources of constraints on lexical choice

2.4 Summary

The sense of an utterance is quite dependent upon its situation of enunciation. Therefore, the surface realization choices of a generator should be constrained by a model of that situation. There are three major aspects to that situation: encyclopedic, interpersonal and discursive. Five sources of knowledge should thus be available to constrain lexical choices in a generator modeling the situation: the grammar, the lexicon, the model of the encyclopedic situation, the model of the interpersonal situation and the model of the discursive situation.

3. Grammatical constraints

3.1 Introduction

This section surveys grammatical constraints on lexical choice. Numerous open-class lexical items have special grammatical properties. Examples of such properties are given in figure 3-1. These properties restrict the possible grammatical forms of the sentences they are used in. Some of these properties depend on the semantics of the lexical item (e.g. adjectives expressing the "obviousness" of something, such as *apparent, clear, evident, obvious* all require *that* before their complements). However, some others do not. They are idiosyncratic properties unpredictable from a semantic classification (e.g. the property for an adjective to be used both as modifier and as complement). Thus, different words or phrasal patterns for expressing the same content unit can interfere differently with the grammatical form of the sentence they are used in. For example consider the following sentences:
(1) *Steve owns this beautiful house*
(2) *Steve has this beautiful house*
(3) *This beautiful house is owned by Steve*
* (4) *This beautiful house is had by Steve*

Although *to have* can substitute for *to own* in (1), yielding (2), it cannot substitute for *to own* in (3), because when carrying an ownership sense *to have* cannot be used at the passive voice. Thus lexical choices and grammatical choices constrain each other. How these bi-directional constraints have been implemented in existing generators is now presented.

Some verbs, constrain the indirect object of a clause to occur after the direct object[16]:
Fatima dances mbalax for Steve
* *Fatima dances Steve mbalax*

Some verbs cannot be used at the passive voice:
Steve weighs three hundred pounds
* *Three hundred pounds are weighed by Steve*

Some verbs, allow the ergative transformation with some type of direct objects, but do not with some other types:
She broke the glass --> *The glass broke*
She broke the law --> * *The law broke*

Some adjectives can be used as complements but not as modifiers:
All the students fell asleep
* *The asleep students failed the class*

Some adjectives require a *that* before their complement:
It is obvious that Steve is gone now
* *It is obvious for Steve to be gone now*

Figure 3-1: Examples of grammatical properties of lexical items

[16]In this figure as well as in the remainder of the survey, incorrect sentences are starred.

3.2 Danlos' generator

[Danlos 86] presents an NLG model and its application to the generation of newspaper-like reports of terrorist attacks. Her generator produces 1 or 2 sentence long reports in either English or French from a frame-based representation of an attack. Figure 3-2 gives two reports generated by Danlos' generator from the same input representation. In this domain, the predicative concepts are attacking actions, events resulting from them and motion actions. The non-predicative concepts are persons, locations and times. In Danlos' model, tokens of predicative concepts are mapped onto clause patterns stored in a phrasal lexicon and tokens of non-predicative concepts are mapped onto NPs by special-purpose LISP procedures. These procedures embed the grammatical properties of the nouns, adjectives and participles used to construct the NPs. The remainder of the grammatical constraints on lexical choice is embedded in two declarative knowledge structures: a **lexicon-grammar** and a **discourse grammar**. We describe each of these structure before outlining the generation algorithm using them.

A policeman was killed and four others wounded today in Paris.
Anarchists exploded a remote controlled bomb under the truck in which they were going from their office to a restaurant.

Today in Paris, anarchists killed a policeman and wounded four others by exploding a remote controlled bomb under the truck in which they were going from their office to a restaurant.

Figure 3-2: Example reports generated by Danlos' system

3.2.1 The lexicon-grammar

The lexicon-grammar is the phrasal lexicon of Danlos' generator. An entry of the lexicon-grammar is a simple clause pattern. A simple clause pattern is a partial realization of a predicative concept token. It specifies the lexicalization of the predicate and has one variable for each of its nuclear roles. Figure 3-3 gives examples of lexicon-grammar clause patterns.

The lexicon-grammar formalism is due to the linguist Maurice Gross[17]. He distinguishes between three types of clause patterns: **simple verb clause patterns**, **frozen clause patterns** and **support verb clause patterns**. Each one corresponds to a different mapping between the predicate-argument structure of a

```
(1) N0 respect N1                  (5) N0 make N1 respectful of N2
(2) N0 have respect for N1         (6) N0 become respectful for N1
(3) N0 be respectful of N1         (7) N0 lose respect for N2
(4) N0 give N1 respect for N2      (8) N0 threw the book at N1
```

Figure 3-3: Examples of lexicon-grammar clause patterns

[17] see [Gross 75], [Gross 84], [Gross 86].

concept token and the structure of the clause realizing it. The structuring units of the clause are the subject, the verb, the objects, the complements and the adjuncts. In simple verb clause patterns such as (1), the predicate is realized by the verb. In frozen clause patterns such as (8) the predicate is realized by an idiomatic expression. In support verb clause patterns such as (2)-(7) the predicate is realized by a noun or an adjective *supported* by a verb. For example, the predicate realized in (1) by the verb *to respect*, is realized in (2) by the noun *respect* supported by the verb *to have*. The presence of support verbs is necessary because the grammar of English[18] requires every clause to have a main verb. But when used as a support, the meaning of a verb is neutralized. This notion of support verb allows stressing the similar semantic structure of clauses with different syntactic structures. Consider, for example, clause patterns (1), (2) and (3). They share the same semantic structure. This structure can be represented by the predicate-calculus formula RESPECT(N1,N2). But while N1 is a direct-object in (1), it is a complement in (2) and (3).

For each clause pattern, the lexicon-grammar gives the values of a set of binary properties. These binary properties express restrictions on both the grammatical form in which the pattern can be instantiated (e.g. *active-voice-only*) and on the types of each role fillers (e.g. *agent-human-only*). Although most of these constraints are purely syntactic (e.g. *active-voice-only*) some are selectional restrictions (e.g. *agent-human-only*). A lexicon-grammar can be represented by a binary matrix with one line per clause pattern and one column per property. An small portion of the lexicon-grammar used by Danlos' report generator is given in figure 3-4.

Insert here figure 9. on p.143 of Danlos book

Figure 3-4: Small subsquare of Danlos' generator lexicon-grammar (from [Danlos 86])

The team directed by Maurice Gross at the LADL[19] made a systematic study of large corpora of real texts. They determined that about 57,000 clause patterns suffice to cover most common-use non-technical French, and around 600 binary properties are necessary to completely describe their grammatical behaviors. These figures mean that a 57,000x600 binary matrix is large enough to store the grammatical properties of most clause patterns of a natural language. This is a very significant result supporting the claim that

[18]and of most Indo-European languages as well.

[19]Laboratoire d'Automatique Documentaire et Linguistique.

mapping predicative content units onto clause patterns such as those of Gross' lexicon-grammar is the right level of clause microcoding for a generator. The on-tape availability of the French lexicon-grammar compiled at the LADL combined with Danlos' demonstration that it can be profitably used in a generator, makes it an interesting source of knowledge for builders of **Natural Language Processing (NLP)** systems in French. Some portions of a lexicon-grammar of English have also being compiled (cf. [Salkoff 83], [Freckelton 84]). From an NLG perspective, the lexicon-grammar formalism has two major drawbacks:

- By not supporting the representation of other **Surface Linguistic Units (SLUs**) than clause patterns (NP patterns, nouns, adjectives etc) it precludes a uniform representation of the grammatical properties of all SLUs.
- By using an *unstructured* set of *binary* features, it precludes a hierarchical organization of the lexicon reflecting the conceptual hierarchy of the encyclopedic knowledge base, as in PENMAN for example (cf. next section).

3.2.2 The discourse grammar

The terrorist attack domain chosen by Danlos emphasizes the expression of causality. The short texts generated by Danlos' generator must not only describe an attack and its result, but also express the direct causal relationship between the two. The expression of causality is challenging for a generator because whether a given text expresses a causal relationship depends on several types of generation choices. In the particular case of Danlos' generator these choices are the following:

- *Number of sentence*: To be chosen between one or two. In Danlos' generator the token representing the attack and the token representing its result are each mapped onto a simple clause pattern. The choice of number of sentences is thus a choice of embedding one clause into the other vs. juxtaposing them next to the other.
- *Sentence ordering choice*: In case juxtaposition is chosen, which sentence should be put first.
- *Voice choice*: For each simple clause either active or passive.
- *Ellipsis choice*: In case the passive voice is chosen for a clause, whether to express or to omit the agent of the clause.

A **discourse grammar** is the exhaustive list of all choice combinations resulting in texts satisfactorily expressing the causal relationship between the attack and its result. For combinations resulting in embedded clause structures, the grammatical type of the embedded clause (e.g. coordinated participial clause, subordinated participial clause, relative clause) is also indicated. The discourse grammar of Danlos' generator (with an example text for each choice combination) is given in figure 3-5. Among the 26 possible choice combinations, only 15 lead to the production of text unambiguously expressing a direct causal relationship. Such a constraint is linguistic data that has to be encoded in some way for a generator to be able to avoid incorrect and awkward texts. That type of constraint is characteristic of a whole class of constraints involving factors of different nature and whose scope extends beyond the sentence level. These constraints are not compiled in any natural language grammar and the process of their enforcement by native speakers is unconscious. This observation led Danlos to claim that the only way for a generator to be able to enforce such constraints is to be provided with a discourse grammar. Discourse grammars are constructed by hand at knowledge acquisition time. This task was manageable for Danlos because the number of generation choices and the number of alternatives for each of these choices is very small in her generator. For example there are only two possible sentence orderings because:

- The length of the generated text is limited to two sentences.
- Each sentence expresses one of only two possible content types (either the attack or its result).

Insert here figure 5. on p.90 of Danlos book

Figure 3-5: Danlos' generator discourse grammar (from [Danlos 86])

For the generation of significantly longer texts whose sentences express a significantly greater variety of contents, writing a discourse grammar is a task of combinatorially explosive size. In other words the discourse grammar approach does not scale up. Its applicability is thus limited to the implementation of *local* constraints involving a small number of factors such as direct causality. Whether such an implementation could be integrated with existing techniques to represent *global* constraints in text generators (e.g. schemas [McKeown 85b], text-spans [Mann & Thompson 87]) is an open question. The alternative approach to such an integration, is to discover rules that predict the acceptability of a text resulting from a given generation choice combination. Danlos claims that the search for such rules is a dead-end. For example she challenges future researchers to find rules explaining the interaction of focus (cf. section 7.1 for a discussion of focus) constraints with any other constraint. Accepting this claim amounts to denying the feasibility of text generation in semantically rich domains. Needless to say, this claim has not gained common acceptance although no one has yet been able to prove it wrong.

3.2.3 Danlos' generation algorithm

Now that we have seen the two main knowledge structures of Danlos' generator, we can present the algorithm it follows.

1. *Choice in the lexicon-grammar of the clause patterns realizing the predicative concept tokens of the input*. As we have seen, these tokens are the attacking act, its result and possibly the motion act of the target at the moment of the attack. These choices are solely based upon encyclopedic constraints and thus are presented in section 5.2.

2. *Instantiation of the clause patterns*, i.e. filling of the placeholders in the clause patterns by the role values of the predicative concept tokens they express (e.g. if the clause pattern *N0 shoot N1* has been chosen to express the token ATTACK-01 of the concept ATTACK: instantiation of N0 by the value of the AGENT role of ATTACK-01 and instantiation of N1 by the value of the TARGET role of ATTACK-01).

3. *Choice in the discourse grammar of a discourse structure compatible with the grammatical, encyclopedic and textual organization constraints induced by clause patterns chosen in step 2*. For example, if the clause pattern chosen to express the attack cannot be passivized (as indicated in the lexicon-grammar entry for that pattern), then a discourse structure with an active clause for the attack is picked. If only one discourse structure is compatible with these constraints it is chosen. If several are, one is chosen randomly. If none are, a discourse structure satisfying only a subset of these constraints is chosen (clause pattern choices are never questioned).

4. *Expansion of the discourse structure into a text template*, i.e. application of the transformations associated with the chosen discourse structure to the chosen clause patterns (e.g. if the discourse structure specifies that a single sentence must express both the attack and its result, the clause pattern expressing the result is subordinated to the clause pattern expressing the attack).

5. *Choice of the noun phrases realizing the non-predicative concept tokens filling the placeholders of the instantiated clause patterns*. As we have seen, these tokens are the agent of the attack, its target and possibly its spatial and temporal location. These choices are solely solely based upon encyclopedic constraints and thus are presented in section 5.2.

6. *Application of morpho-syntactic rules to the fully lexicalized text template* (e.g. application of the verb-subject agreement rule).

3.2.4 Representation of grammatical constraints in Danlos' model

Danlos' generator takes into account three types of grammatical constraints:

1. The grammatical constraints on the expression of direct causal relationship between two clauses.

2. The grammatical properties of the clause patterns used to realize the predicative concept of the application domain.
3. The grammatical properties of the words used in the noun phrases realizing the non-predicative concept of the application domain.

Because it is hand-coded in its discourse grammar, Danlos' generator is able to enforce the first type of constraints with more rigor than any other NLG systems. In these systems, this type of constraints is desemphasized by the application domain. In Danlos' generator this type of constraint is integrated in the discourse grammar. The main drawback of discourse grammars are their heterogeneity. Heterogeneous knowledge structures integrating different type of constraints cannot be used by modular systems where different types of constraints are dealt with by different modules. However, modularity remains the only hope for any form of portability over application domains. The second type of constraint is embedded in the lexicon-grammar. We discussed it in section 3.1.1. The third type of constraint is scattered in the LISP procedures implementing the realization of noun phrases. From a software engineering perspective, this approach carries all the software engineering weaknesses of non-modular procedural knowledge representation. Overall, the main problem with Danlos' combination of representation schemes is that it lacks the uniformity necessary to achieve extensibility.

3.3 PENMAN

3.3.1 An overview of PENMAN

Almost all complete text generators are special-purpose systems developed with a particular application in mind. Each specific application requires tackling some hard generation problems. It also allows the designers to make simplifying assumptions for most others. Consequently, the portability of the components of such application-oriented systems is often limited. **PENMAN** [Mann 82] is the first large-scale attempt to address the issue of portability in NLG. It is a long-term project to build a domain-independent tool-kit for NLG applications. Among the various types of knowledge needed in NLG, grammatical knowledge is both the best understood and the least variable across applications. The grammar is thus the best candidate for complete portability among generation components. The development of PENMAN was therefore initiated by the development of the **NIGEL** [Mann & Matthiessen 83] grammatical component.

In its current implementation, NIGEL is able to output almost all forms of English sentences. Its coverage is thus more extensive than any other generation grammar. NIGEL makes use of a systemic grammar[20]. The systemic formalism is particularly well-suited for generation because it views:

- The grammar as a set of expressive resources.
- The grammatical form of a sentence as resulting from choices among these resources.

A systemic grammar is structured as a network of "systems". Each system represents a set of alternative grammatical features to choose from (e.g. *definite/indefinite* for an NP). The choice of an alternative in a given system leads to the entrance of another system, i.e. to encounter another alternative (e.g. choosing *indefinite* in the *definite/indefinite* alternative leads to the *singular/plural* alternative). The generation of a sentence is controlled by the traversal of the systems network. During this traversal, each time a choice

[20]see [Halliday 76] for a comprehensive introduction to systemic grammars and [Winograd 83] for a quick and pedagogical one.

point is encountered, NIGEL needs information to perform the choice and proceed to the next one. This information has to be provided by some sources external to the grammar. Thus, the primary issue raised by the development of NIGEL is to identify such information sources. It is the converse of the issue raised by the development of a grammar for an application-specific generator (namely identifying the grammatical forms commonly used to express given information). Since the development of a domain-independent NLG tool-kit cannot be constrained by the application, it can only be constrained by a specific natural language (English for PENMAN). In addition to implementating a grammar, interfaces between the grammar and various choice-information sources must be defined. The first interface to NIGEL developed in the PENMAN tool-kit was the use of **inquiries**. Inquiries are multiple-choice questions prompted to the user whenever NIGEL encounters a choice point. They allow the user to provide all the information NIGEL needs to output a sentence. The current architecture of the PENMAN tool-kit is given in figure 3-6.

[Matthiessen 87] classified the information needed by a surface realization component[21] using the three

Insert here figure 1 page 12 of PENMAN user guide

Figure 3-6: Architecture of the PENMAN tool-kit (from [PENMAN 88])

[21] i.e. a grammatical chooser together with a lexical chooser.

grammatical meta-functions put forward by [Halliday 85]: ideational, interpersonal and textual They roughly correspond to our encyclopedic, interpersonal and discursive situations of enunciation. We have seen in section 2.3 that there are four types of external constraint sources on lexical choices: grammatical, encyclopedic, interpersonal and discursive. Similarly there are also four types of external constraints sources on grammatical choices: lexical, encyclopedic, interpersonal and discursive. Each component of the PENMAN kit is an interfacing tool to one of these four constraint sources: the **MasterLexicon** [Cumming & Albano 86] to the lexical constraint sources, the **Rhetorical Structure Theory (RST)** [Mann & Thompson 87] to the discursive constraint sources, the **Upper-model** and the **Sentence Planning Language (SPL)** [] to the encyclopedic constraints sources. No interfacing tool to interpersonal constraint sources is yet provided. From the perspective of the chapter, namely the representation of grammatical constraints on lexical choice, only two of these tools are relevant: the MasterLexicon and the Upper-model.

3.3.2 Lexical choice using PENMAN

PENMAN provides a framework to build an **Encyclopedic Knowledge Base (EKB)**: the knowledge representation system **LOOM** [Mac Gregor & Bates 87] and the Upper-model. It also provides a framework to represent the grammatical properties of lexical items called the MasterLexicon [Cumming & Albano 86]. However, PENMAN does *not* provide a framework to perform the mapping between the two, i.e. lexical choice. A lexical chooser *could* be integrated into a generator built using the PENMAN tool-kit. But both the lexical chooser and its interfaces to the other components of PENMAN would have to be written from scratch. However, both the EKB Upper-model and the MasterLexicon provide minimal but useful guidance for this task.

The Upper-model is a portable upper-part for hierarchical EKBs. It captures some essential ontological distinctions between real-world entities which are grammatically marked. For example the distinction between divisible and undivisible entities is reflected in English by the distinction between mass and count nouns. It is thus important for these grammatically-marked ontological distinctions to be explicit in the hierarchical structure of the EKB modeling the ESE. The Upper-model includes the high-level concepts representing these distinctions. For each application, the domain-model concepts are hooked-up below the Upper-model concepts. This way, the overall EKB for an application (Upper-model + domain-model) is ensured to contain the distinctions needed by NIGEL to perform grammatical choice.

The MasterLexicon provides a set of features to represent the grammatical properties of lexical items. To express the dependencies between features (e.g. the *transitivity* feature depends on the *POS* feature since it is relevant only for verbs), they are organized into a **word-class hierarchy**. Moreover, the structure of this hierarchy reflects the structure of the Upper-model (e.g. the feature *perception* for a verb corresponds to the Upper-model concept PERCEPTION representing the class of perception processes). This coherence between grammatical features and EKB concepts allows the grammatical properties of lexical item to be viewed as resources to express semantic content. This view is essential to use these properties as constraints on lexical choice. Another quality of the MasterLexicon's feature list is that it is fairly exhaustive.

The lack of lexical chooser is definitely the major gap in the PENMAN tool-kit. However, although it ignores every constraint on lexical choice other than grammatical, the PENMAN project put forward for the implementation of grammatical constraints an interesting methodology and some invaluable data . In particular, the coherence between grammatical and encyclopedic hierarchies advantageously compares with the lexicon-grammar approach of Gross. How harmoniously the PENMAN representation of grammatical constraints could be integrated to other constraints representation on lexical choice remains to be seen.

More important, the first run of an NLG application built using the PENMAN tool-kit also remains to be seen.

3.4 Other generators

Except for Danlos' generator and PENMAN, other generators disemphasize the representation of open-class words' grammatical properties. In these systems, the representation of idiosyncratic properties is largely avoided by not attempting to generate utterances in which they manifest themselves, and the representation of general properties is embedded in the grammar.

3.5 Summary

Two research efforts, one by Danlos' and the other by the PENMAN group emphasizes the idiosyncratic grammatical properties of lexical items.

The motivation underlying Danlos' emphasis on lexical items grammatical properties is the specificity of her domain. Her domain requires the expression of direct causal relationships and such an expression results from the complex interaction of several constraints, including grammatical constraints on lexical choice. Danlos' generator uses a discourse grammar, an exhaustive list of generation choice combinations resulting in texts satisfactorily expressing the desired direct causal relationships. The use of a discourse grammar makes Danlos' generator very robust[22] but hardly extendible.

The motivation underlying the emphasis on grammatical properties lexical items in PENMAN is the opposite of Danlos': the deliberate lack of any domain-specific restrictions. Because the grammar is the only generation component for which total portability can be achieved, PENMAN is a tool-kit built around NIGEL, an extensive coverage stand-alone grammatical chooser. The other tools of the PENMAN kit are representation systems for the implementation of generation application components providing NIGEL with the information it needs to perform grammatical choice. These tools come with the bootstrapping knowledge necessary to interface these components with NIGEL. The bootstrapping knowledge provided for the building of an EKB and the bootstrapping knowledge provided for the building of a word-based lexicon are isomorphic hierarchies. Although, this isomorphism is convenient because it allows viewing both the lexicon and the grammar as expressive resources, PENMAN does not provide any representation system to implement lexical choice.

[22]robust in its ability to avoid generating incorrect or awkward texts.

4. Interlexical constraints

This section surveys mutual constraints between choices of different words. In a microcoded generator, the realization of a sentence involves several open-class word choices. These different choices are generally interdependent. The choice of one open-class word can put additional constraints on the selection of the other open-class words of the same sentence. This is the case when a particular type of lexical relation, collocation, holds between the candidate lexicalization candidates of two tokens in a content unit. We first characterize collocations amidst lexical relations. We then survey how collocations are used to constrain lexical choice in various generators.

4.1 Lexical relations

Open-class words can be involved in many different types of lexical relations. Several distinctions have to be made to identify these different types. The first distinction is between **syntagmatic** and **paradigmatic** relations. Two words are syntagmatically related if they appear close to each other in correct phrases. Two words are paradigmatically related if they appear in place of each other in correct phrases. Consider the example phrases of figure 4-1. In (1) *old* and *man* are syntagmatically related just as *wonderful* and *door* are in (2). Replacing *old* by *wonderful* in (1) yields (3) a correct phrase. Thus *old* and *wonderful* are paradigmatically related. Similarly, *man* and *door* are paradigmatically related. However, the incorrectness of (5) and (6) indicates that neither (*wonderful*,*old*) nor (*man*,*door*) are syntagmatic relations. It also indicates that neither (*wonderful*,*man*) nor (*old*,*door*) are paradigmatic relations. Due to [Saussure 16] the syntagmatic/paradigmatic distinction is based on the view of natural language utterances as resulting from two processes: a process of combination and a process of selection. Within this view, the generation of an utterance consists in combining along the syntagmatic axis words selected along the paradigmatic axis. Figure 4-1, shows the locations of *old, man, wonderful* and *door* on the plan defined by these two axis.

Classes of lexical items underlying paradigmatic relations are classes of lexical items that can be substituted one in place of one another. From sets of words used in a given encyclopedic domain (e.g. medicine, stock market, transportation etc) and binary relations (e.g. antonymy, synonymy, meronymy), these classes have various degrees of generality. However they are always encyclopedically grounded. Hence, paradigmatic relations between lexical items are inherently *indirect* via the concepts these items respectively lexicalize. In a generator, the relations between concepts have to be represented in the EKB

```
                             Paradigmatic
                                Axis
                                  ^
                                  |
(1) The old man                   |
(2) The wonderful door            |
(3) The wonderful man             |  old         man
(4) The old door                  |
(5) * The wonderful old           |  wonderful   door
(6) * The man door                |
                                  |----------------------> Syntagmatic Axis
```

Figure 4-1: Syntagmatic and paradigmatic axis and the correspondence between these concepts and lexical items also has to be specified[23]. Therefore, the representation of paradigmatic relations between lexical items is best achieved via the representation of the encyclopedic relation holding between their respective associated concepts.

In contrast to paradigmatic relations, syntagmatic relations are *direct* relations between lexical items, and as such they should be explicitly represented in a generator. Grammar is a decisive constraining factor on combinations along the syntagmatic axis. To come back to our example phrases: the incorrectness of (5) is simply due to the fact that adjective-adjective modification with no head noun is ungrammatical in English. Encyclopedic factors also constrain syntagmatic combinations: although noun-noun modification is grammatical, (6) is incorrect because the combination *man door* does not correspond to any existing concept.

(7) *The door man*

on the other hand is correct since the combination *door man* lexicalizes the concept of a man which job is to keep a door. Selectional restrictions are another example of encyclopedic constraints along the syntagmatic axis. For example the requirement of an animate agent for the concept of breathing explains the correctness of:

(8) *The woman breathes*

and incorrectness of:

(9) * *The door breathes*

However, as pointed out by [Halliday 76], grammar and encyclopedia are not the only factors constraining the combinations along the syntagmatic axis. Consider the following phrases:

(8) *The powerful car*

(9) *The strong tea*

(10) * *The powerful tea*

(11) * *The strong car*

Powerful and *strong* belong in the same grammatical category. In the above sentences, they also realize the same grammatical function and lexicalize the same concept. The fact that *powerful* has to be used in conjunction with *car* and *strong* with *tea* is thus unpredictable upon grammatical or encyclopedic grounds. It is a purely interlexical constraint.

Support verbs (i.e. verbs used to support the lexicalization of a predicative concept by a noun or an adjective) are another example of purely interlexical constraints. As we have seen in section 2.1.1, when used as a support, the meaning of a verb is neutralized its presence being only motivated by the obligation for every clause to have a main verb. Thus from an encyclopedic perspective, its choice is arbitrary; the encyclopedic constraints (i.e. the token to realize) dictate only the choice of the noun or adjective it supports. Consider phrases (12)-(14) of figure 4-2. In the semantics of the process expressed in (12) is exclusively carried by the meaning of *question*. The constraint to use *to ask* as a support verb with *question* is purely interlexical and independent of the meaning of *to ask*. This semantic arbitrariness of support verb choice is illustrated by its language-dependence. In French, the support verb for *question* (*question*) is *poser* (*to put*) and in Turkish the support verb for *soru* (*question*) is *sormak* (*to question*). The neutralization of the meaning of a verb when used as a support is also well illustrated by the versatility of the serbo-croatian verb *pustiti*. This verb is used to support lexicalizations of completely unrelated concepts, as showed by phrases (15)-(20) of figure 4-2.

In the literature, relations such as (*tea,strong*), (*question,to ask*) or (*zmaja,pustiti*) have been called

[23]Either declaratively in the lexicon or procedurally in the lexicon interface.

```
     English                   French                        Turkish

12: I asked a question       *J’ais demande une question    *Bir soru istedim
13: *I put a question        J’ais pose une question        *Bir soru koydum
14: *I questioned a question  *J’ais questione une question  Bir soru sordum


    Serbo-Croatian      English

15: pustiti bradu       to grow a beard
16: pustiti koren       to take root
17: pustiti krv         to draw blood
18: pustiti motor       to start an engine
19: pustiti plocu       to play a record
20: pustiti zmaja       to fly a kite
```

Figure 4-2: Language-dependence of support verbs

restricted lexical co-occurrences, or **collocations**. They are *constrained* syntagmatic relations, unpredictable from grammatical or encyclopedic factors. These relations are activated in a given grammatico-encyclopedic context, but they put additional constrain on lexical choice. Their unpredicability sets collocations in contrast to **free-combinations** such as *wonderful man*, *old door* or *woman breathes* which are constrained by the grammatico-encyclopedic context *only*. Free-combinations are transparent surface linguistic units. Their meaning can completely be computed from the meaning of the words composing them. On the other hand, idioms (e.g. *to kick the bucket*, *to throw the book at*) are opaque surface linguistic units. Their meaning cannot be computed from the meaning of the words composing them. In terms of opacity, collocations lies in between free-combinations and idioms.

Collocations have to be explicitly represented in a generator to avoid outputing phrases like (10), (11), (13) or (14). The representation of collocations differs significantly in macrocoded and microcoded generators. We review the representation of collocations in each of them in turn.

4.2 Collocations in macrocoded generators

In macrocoded generators (e.g. Danlos’ generator [Danlos 86], ANA [Kukich 83], PHRED [Jacobs 85]) collocations are hard-wired in the phrasal patterns of the lexicon. For example, in ANA, both collocations (*mood, hesitant*) and (*mood, display*) are hard-wired in the clause pattern: *X display an hesitant mood for most of the day*. In these generators, members of a collocation relation are all selected at once by the selection of a phrasal pattern. The macrocoding approach carries with it inherent limitations in terms of variety of generable SLUs. For example, using the phrasal patterns:
N0 respect N1
N0 have respect for N1
N0 be respectful of N1
NO show respect for N1
to lexicalize the concept RESPECT(N0,N1), does not allows the generation of clauses like:
(21) *His teammates show lasting respect for Steve*
(22) *His teammates always had the greatest respect for Steve*
containing modifier constituents not included in these patterns.
One possible solution to overcome such limitations is to microcode the modifiers in these clause patterns, yielding:

N0 respect N1 M0
N0 have M0 respect for N1
N0 be M0 respectful of N1
NO show M0 respect for N1
But with such patterns, part of the hard-wiring of collocations is lost and the avoidance of incorrect clauses such as:
(23) * *His teammates respect Steve lastingly*
(24) * *His teammates were greatestly respectful of Steve*
can only be enforced by explicitly representing collocational knowledge. For very simple domains such as those of Danlos' generator and ANA, the simplification allowed by the macrocoding approach advantageously balances with the limitations on surface form variety. However, for more ambitious applications such a variety is required, and the only practical option is microcoding.

4.3 Collocations in microcoded generators

As illustrated by example sentences (23) and (24) above, enforcing collocational correctness while microcoding surface linguistic units requires the explicit representation of collocational knowledge. How this knowledge has been acquired, represented and used in existing microcoded generators is now surveyed.

4.3.1 Meaning-text theory report generators

A team of computational linguists of University of Montreal and Odyssey Research Associates have developed report generators in several domains: stock market (SMRAD [Kittredge & Mel'cuk 83]), operating systems audit (GOSSIP [Carcagno & Iordanskaja 89]) and meteorology (RAREAS [Kittredge, Polguere & Goldberg 86]). An example of generated text in the meteorology domain is given in figure 4-3. They take as input raw quantitative data and output texts summarizing them. In these generators, the modules responsible for surface realization are based on the **Meaning-Text Theory (MTT)** [Mel'cuk 81].

The MTT is a generation-oriented and lexicon-centered grammatical formalism. The formalism is itself structured as a pipeline. Each element in the pipeline specifies the mapping from the internal representation of a sentence at a given level to the internal representation of the sentence at the next level. Each mapping is encoded in a set of rules[24], embedding general grammatical and morphological knowledge. The more lexically related knowledge necessary for each mapping is encoded in the **Explanatory Combinatorial Dictionary (ECD)**[25]. An ECD entry is divided in three zones:

- The **semantic zone**: a semantic network representing the meaning of the entry in terms of more primitive words. The deepest internal representation of sentences in the MTT uses the same type of semantic network.
- The **syntactic zone**: the set of grammatical properties of the entry (i.e. the properties we discussed in section 3).
- The **Lexical Combinatorics Zone (LCZ)** containing the values of the **Lexical Functions (LFs)** for that entry.

[24] [Mel'cuk & Pertsov 87] give the rules for an MTT model of English.

[25]see [Mel'cuk & Polguere 87] for an NLG-oriented presentation of the ECD.

Frobisher-bay
Gale warning issued...
Winds light becoming southeasterly 15 early Tuesday morning then backing and strengthening to easterly 30 Tuesday afternoon then strengthening to northeasterly gales 35 Tuesday evening. Mostly cloudy with snow, fog and mist patches. Visibility fair in snow, fair in mist and poor in fog. Outlook for Wednesday ... gale force northeasterlies becoming gale force northerlies

Figure 4-3: A weather forecast generated by RAREAS

A lexical function maps a word onto a list of words. [Mel'cuk 82] claims that the 50 lexical functions he provides allow the representation of paradigmatic lexical relations and collocations involved in the paraphrasing power of any Indo-European language[26]. Examples of lexical functions representing collocations are given in figure 4-4.

```
Sing(X)="element of a regular set of X"   Mult(X)="regular set of Xs"
Sing(swarm)={bee,locust,sting,...}        Mult(bee)=swarm
Sing(advice)={bit,word,piece}             Mult(ship)=fleet

Oper1(X)="support verb of X, when X in direct object position, and the
agent of the action lexicalized by the verb is in subject position"
Oper1(question)=to ask          Oper1(advice)=to give
Oper1(attention)=to pay         Oper1(cry)=to let out

Magn(X)="That intensifies X"    Son(X)="Typical sound emitted by X"
Magn(escape)=narrow             Son(lion)=roar
Magn(bleed)=profusely           Son(floor)=creak
```

Figure 4-4: Examples of lexical functions

Let us now see how these LFs are used to constrain lexical choices on an example. Consider the generation of the following sentences:
(25) *The Food and Drug Administration has seriously cautioned expectant mothers to avoid one of life's simple pleasures: a cup of coffee*.
(26) *Pregnant woman have been earnestly warned by the FDA against drinking coffee, one of the small pleasures of life*.
(25) and (26) are two alternative realizations of the same token, say WARNING-01, of the WARNING concept with a HIGH value for the INTENSITY role. In a generator using the MTT as surface component, the realization of this token proceeds as follows:

1. Choice of the main verb of the clause to express that WARNING-01 is an instance of WARNING (respectively *to caution* for (25) and *to warn* for (26)).
2. Choice of the LF **Magn** to express the HIGH value for the INTENSITY role of WARNING-01.

[26]lexical functions can be composed, the 50 lexical function Mel'cuk proposes are thus building blocks to describe lexical relations.

3. Choice of the adverb modifying the main verb by looking-up in the ECD the value of **Magn** for that verb (respectively *seriously* for *to caution* and *earnestly* for *to warn*).

Lexical functions are an interesting formalism to represent collocations because they classify collocations in terms of both the grammatical and the encyclopedic functions they realize. However, using lexical functions to represent collocations when building a microcoded generator means adopting one of the following alternatives: either adopting the MTT framework as a whole or trying to integrate lexical functions with another surface generation framework. There are serious problems with both options of this alternative. With the first options the problems are the following:

- The published literature on MTT generators emphasizes the presentation of the theory itself and neglects discussing its practical use to implement surface realization for an NLG application.
- [Carcagno & Iordanskaja 89] give some clues on how to link a textual organization component to an MTT surface realization component, but they say nothing about the mapping between content units and surface linguistic units. As input, an MTT surface realization component takes a semantic network whose nodes are words (the common MTT input for sentences (25) and (26) is given in figure 4-5). The entities of the underlying application have to be mapped onto such word-based networks. How this mapping is done or could be done remains unclear.
- We have seen in the introduction of this survey that NLG consists of two distinct subtasks: (1) identifying all the possible ways to express a given content and (2) choosing among them the most appropriate to a particular situation of enunciation. [Boyer & Lapalme 85] convincingly argue that MTT is a powerful framework for the first of these subtasks. But, MTT largely ignores the second one. In particular, how to use models of the ESE, DSE and ISE to *choose* between the surface realization options provided by MTT remains unclear.
- An MTT surface realization component heavily relies on an ECD. No ECD of English is available. An ECD of French is incrementally being published [Mel'cuk & al. 84] but, at the present time, less than 20 words have been covered ! Besides, the MTT literature says very little about methodology for compiling ECDs.

With the second option the problems are the following:

- Numerous lexical functions definitions are vague and intuitive. In particular, encyclopedically grounded lexical functions are not even defined in terms of MTT semantic networks, but by natural language paraphrase.
- The values of the lexical functions for every word have not been compiled in any systematic way.

Thus, using LFs in a generator requires to compile these values by hand with, for sole guidance, the few illustrative examples given in the MTT literature. In short, although the notion of lexical function in itself is appealing for the representation collocations in a microcoded generator, using the very LFs proposed by Mel'cuk is unpractical without the collaboration of an MTT guru.

Insert here figure 1 p.18 of Mel’cuk & Pertsov

Figure 4-5: MTT input[27] for sentences (25) and (26) (from [Mel’cuk & Pertsov 87])

4.3.2 Other generators

DIOGENES ([McCardell 88] and [Nirenburg 89]), the generation component of the TRANSLATOR [Nirenburg 87] machine translation system, is another attempt to constrain lexical choice using lexical relations. In DIOGENES' lexicon an entry has two parts: a conceptual pattern and a linguistic description. Both parts are represented using a frame-based language. The conceptual pattern specifies the concepts the entry can lexicalize. The linguistic description includes some grammatical properties of the entry as well as cross-references to other entries. From the example entries presented in [McCardell 88] (one of them is given in figure 4-6) it seems that the lexical relations represented by these cross-references includes many syntagmatic free-combinations (e.g. (*boy*,*playground*), (*boy*,*pen*)) and paradigmatic relations (e.g. (*boy*,*girl*) (*boy*,*men*)) but few if any real collocations. For the purpose of performing lexical choice, what can be gained by storing free-combinations is unclear. Moreover, as we saw in section 4.1, the representation of paradigmatic relations between lexical items is best achieved via the representation of the encyclopedic relation holding between their respective associated concepts. In contrast paradigmatic relations are represented in DIOGENES by explicit pointers tagged by labels such as "synonym" or "antonym". Since these labels are not defined in terms of the encyclopedic relations holding between the associated concepts, maintaining them seems to be not only redundant but inconsistency-prone.

```
(((is-token-of (value person))
  (sex (value male)
       (importance 10))
  (age (value (0 12))
       (importance 6)))
 ((english boy)
  (lexical-class noun (noun-type class) (morph regular))
  (collocation (location school playground)
               (agent-of do-homework play-ball)
               (instrument pen notebook ball)
               (set-complement-sex girl)
               (set-complement-age baby man))))
```

Figure 4-6: Lexicon entry for the word *boy* in DIOGENES

Other microcoded generators simply avoid the issue of collocations. In these generators the surface realization component accepts overspecified input where all the elements of a collocation are already wired-in. This can be done by having a one to one mapping between words and EKB concepts and by carefully choosing collocationally compatible concept names at knowledge acquisition time. Naturally, this is not only avoiding lexical choice altogether but it also becomes unmanageable for EKBs of any significant sizes.

4.4 Acquisition of collocational knowledge

Numerous references books (dictionaries, grammars, encyclopedia) are making grammatical and encyclopedic knowledge explicitly available to humans in a systematic way. For this type of knowledge, the knowledge acquisition task thus consists in the conversion of this knowledge from an informal human-targeted format, into a formal format usable by a computer program. In contrast no systematic compilation of explicit collocational knowledge is available, even in human-targeted format. Native speakers use collocational knowledge unconsciously, but the lack of such knowledge accounts for numerous errors of second-language learners (cf. [Leed & Nakhimovsky 79]). We have seen that one of the main drawback of Mel'cuk's lexical functions as a formalism to represent collocational knowledge is the lack of reference book containing the systematic compilation of their values. To the best of our knowledge there are only two reference books on English collocations: "The Word Finder" [Rodale & al. 47] and "The BBI combinatory dictionary" [Benson, Benson & Ilson 86]. For each entry, the Word-Finder simply gives a unstructured list of words (with their Parts-Of-Speech). Among these words there are far more free-combinations than collocations. Moreover, many important collocations are missing. The BBI has the advantage of not containing any free-combinations. For each entry, it gives both the collocates of the entry and the closed-class words used with the entry in specific grammatical constructs[28]. The BBI is an excellent source of knowledge for second-language learners. But from an NLG perspective it has two significant drawbacks:

- Collocations are not tagged by the grammatical and encyclopedic functions they realize. Functional tagging of surface linguistic units is essential for lexical choice.
- Coverage is limited to a relatively small number of words of common usage. Therefore numerous collocations encountered in the domain-specific technical texts most NLG application aim to produce are missing.

This lack of appropriate collocation reference book, led to the development of software tools for the acquisition of collocational knowledge from online texts. Conceptually, acquiring collocations can be decomposed in three subtasks: retrieval, grammatical tagging and encyclopedic tagging. For the (*dream,to have*) collocation for example, grammatical tagging consists in identifying that *dream* preferably co-occur with *to have* when involved in a head-of-direct-object/main-verb grammatical pattern. Encyclopedic tagging consists in identifying that the locution *to have a dream* is one way (another being simply *to dream*) to lexicalize the predicative concept of DREAMING. By its very nature, collocational knowledge can only be acquired from large textual corpora. Given that semantic interpretation of unrestricted texts is far beyond the state-of-the-art in NLU, encyclopedic tagging of collocations cannot be automated. Even grammatical tagging cannot be fully automated since complete structural disambiguation requires some level of semantic interpretation. However, software tools performing collocation retrieval and partial grammatical tagging could be advantageously included to linguistic knowledge acquisition environments.

XTRACT [Smadja 89] is an example of such tool. It retrieves collocations by using brute-force statistics on large textual corpora. XTRACT can compile either domain-independent co-occurrence patterns from very large multiple-domain corpora or domain-specific co-occurrences patterns from large single-domain corpora. From an NLG perspective, this second option is particularly attractive since:

- Computer-generated texts are necessarily domain restricted.
- There is no available reference on domain-specific collocations.

Since based on statistics, XTRACT is sensitive to noise and compiles some free-combinations together with the collocations it is retrieving. Some collocations retrieved by XTRACT, subsequently grammatically and encyclopedically tagged by hand, have been used in **COOK**, a surface generator microcoding simple sentences in a toy financial domain. The collocations have been integrated to the **Functional Unification Grammar (FUG)** (cf. [Kay 79], [Elhadad 88]) used by COOK. They allow COOK to accept input with only one collocation element lexicalized. The other element is lexicalized through unification of its semantic description with the collocation. XTRACT is also able to retrieve some of the domain-specific phrasal patterns found in the phrasal lexicon of macrocoded generators like ANA.

[28] Remember that we classified these relations between open-class and closed-class words among the grammatical properties of open-class items not among collocations.

Co-occurrence pattern compilers have also been developed for computational lexicography (e.g. [Choueka, Klein & Newitz 83], [Church 90]), sentence disambiguation (e.g. [Debili 82]) and information retrieval (e.g. [Maarek 90]). They are all based on either brute-force statistics or partial parsing. There are two possible ways to combine these two techniques for semi-automatic grammatical tagging: either performing statistic analysis on partial parses or performing partial parses of statistically retrieved patterns.

4.5 Summary

Collocations, syntagmatic lexical relations that cannot be accounted on grammatical or encyclopedic grounds, need to be represented explicitly in the word-based lexicons of microcoded generators. Mel'cuk's lexical functions is an appealing formalism for such representation because it classifies collocations in terms of the grammatical and encyclopedic functions they realize. However, lexical functions are difficult to use outside of the Meaning-Text Theory, a grammatical formalism whose ability to represent various types of generation constraints is unproven. Besides, no lexical functions reference book is available. The lack of reference books in collocations lead to the development of software systems that semi-automatically compile collocations from textual corpora.

5. Encyclopedic constraints: world-knowledge

5.1 Introduction

This section surveys encyclopedic constraints on lexical choice. Some constraint types can be ignored by basic forms of generation: single sentence generation can ignore discursive constraints, untailored generation[29] can ignore interpersonal constraints and macrocoded generation can ignore interlexical constraints. But because the content of an utterance is the first of the encyclopedic constraints on its wording, these constraints are as unavoidable as grammatical constraints: if any lexical choice is performed, it must at least be constrained by the content to convey.

Typically, this content consists of a set of propositions about some entities of the underlying application domain. The representation of the domain entails several hierarchies in which each entity is a complex compound of attributes. The propositions expressed by the utterance can concern any aspect of an entity: its position in a given hierarchy, the value of some of its attributes, the type of relation it is involved with another entity, etc. They are extracted from the global representation of the domain by the module(s) responsible for content determination. In an utterance, each of these propositions is expressed by some SLU. Thus, during surface realization, proposition types have to be mapped onto SLU types. The notions of **predicate** and of **roles** are essential for this mapping, because they introduce important distinctions both at the *conceptual* level and at the *linguistic* level.

At the conceptual level, they introduce the distinction between **predicative** and **non-predicative concepts**. From an ontological perspective, this distinction is founded on the fact that, in every domain, some entities are best represented as predicates (e.g. actions, events, processes, relations, states) while others are not[30] (e.g. objects, classes, sets). From a knowledge representation perspective, this distinction can be founded on another one: the distinction between **nuclear roles** and **satellite roles**. The roles of a concept are placeholders for other concepts that can come into play in its descriptions (e.g. the participants of an action, the arguments of a relation, the properties of an object). Restrictions apply on the type of concepts that can fill a given role (e.g. only living entities can fill the *agent* role of a breathing action, only a woman can fill the *mother* role of a human being). The distinction between nuclear and satellite roles is rooted in two factors:

- The nuclear roles of a concept token are a necessary part of (almost) all of its descriptions (e.g. for a donation action, its agent, object and recipient) whereas the satellite roles are part of only specifically detailed ones (e.g. for a donation action, its the location and time)
- Nuclear roles are specific to one concept or to a small class of concepts, whereas satellite roles are pervasive and mainly circumstantial.

Both predicative and non-predicative concepts have satellite roles, but the distinction between the two can be founded on the fact that only predicative concepts have nuclear roles. This fact is supported by philosophers of language like [Mayo 61] who root the ontological distinction they make between objects[31]

---

[29]i.e. generation of the same utterance to convey a given content, regardless of the hearer.

[30]see [Bach 86] and [Bach 86] for a philosophical presentation of the issue.

[31]and sets, classes etc.

and events[32] from the assymetric ways they are referred to in most (if not all) natural languages: only objects can be referred to by proper nouns whereas events can be referred to either by clauses[33] or by their nominalizations as definite descriptions. Moreover in both cases, the reference to the event necessarily includes embedded references to the objects participating in the event. Thus one can say *The coronation of the queen* or *The coronation of Elizabeth* to refer to the event of Queen Elizabeth being coroned, but one cannot say *The queenification of Schtroumpf* to refer to Elizabeth if *Schtroumpf* is the proper noun of her coronation[34] have nuclear roles whereas objects (or more generally what we called non-predicative concept) do not.}.

At the linguistic level, the notions of predicate and roles have also been used to define the grammatical functions of clause constituents ([Quirk & al. 72], [Halliday 85]). The predicate function is generally realized by the verbal group of the clause and the various role functions are realized by the subject, objects, complements and adjuncts of the clause. The constituents of other SLU types (e.g. NP) can also be functionally described in terms of roles. Thus, the realization of a concept token by an SLU involves mapping each **semantic role** of the token into a constituent realizing the **syntactic role** of the SLU. Note that different grammatical forms for a given SLU type may result in different mappings between semantic and syntactic roles. Consider, for example, the realization of a donation act by a clause. With the active voice, the *agent* of a donation act is mapped onto the *subject* of the clause, whereas with the passive voice, it is either mapped onto an *adjunct* or ellided, with the *subject* of the clause then realizing either the *object* or the *recipient* of the donation act.

In most existing generators, most of the difficult issues involved in mapping semantic roles onto syntactic roles are avoided by adopting the following simplifying assumptions:

- The mapping between concept types and SLU types is one-to-one: tokens of predicative concepts are mapped onto clauses and tokens of non-predicative concepts are mapped onto nominals.
- In terms of SLUs types, the variety of clause and nominal constituents realizing syntactic roles is very limited. For example at the clause level, only a restricted set of clause and nominal types can be inserted as clause constituents, and adjectival, adverbial and prepositional phrases are either avoided or hard-wired to clause patterns.

Under these assumptions, the global problem of lexical choice can be decomposed into two subproblems of more manageable size:

- The mapping of predicative concept tokens onto clauses.
- The mapping of non-predicative concept tokens onto nominals.

ALL existing generators have focused on one of them, while adopting a much simplistic scheme for the other. Historically, the generators of the early 80's (cf. [Danlos 86], [Jacobs 85], [Kukich 83]) have been essentially concerned with the first of these subproblems. With the emergence of later works focusing on the second one (cf. [Dale 88], [Reiter 90]), it became clear that each of them is raising a rather different set of issues. In particular, the view of mapping content units onto SLUs as a problem of **reference** was

[32]and processes, actions, states, relations etc.

[33]if clauses are considered referring expressions, see infra.

[34]Although *Elizabeth* or *the Queen* can be ellided if the discursive or interpersonal situation of enunciation conveys that the object of the coronation is Elizabeth, this type of reference by incomplete definite description is quite different from the use of a proper noun. It thus does not invalidate the fact that events (or more generally what we called predicative concepts

immediately prevalent in research efforts focusing on choosing nominals, whereas it was deemphasized in those focusing on choosing clauses. This situation can be explained by several factors:

- From a conceptual perspective, because of the fundamental ontological differences between predicative and non-predicative concepts.
- From a linguistic perspective, although it has occupied linguists, philosophers and logicians for a very long time, the boundaries of the problem of reference remain controversial. Therefore there is no clear consensus on what SLU is or is not a referring expression. Although most nominals are undoubtedly referring expressions, the referring nature of clauses is more controversial.
- From an NLG perspective, because of the very limited types of text NLG systems currently produce, the lexicalization of a predicative concept token tends to occur only once during the production of a text, whereas any paragraph-length generator have to generate chains of successive references to the same non-predicative concept token[35]

Investigating lexical choice from the perspective of reference has several imports. Notably, it stresses the following points:

- The lexicalization of a token can rarely be carried out in isolation. We have seen that, typically, the tokens passed to the lexical chooser are part of a global representation of the underlying domain. In the general case, the lexicalization of a token needs to be constrained by a larger chunk of this global representation than the information contained in the token only. For example, choosing *the giant* to refer to a very tall human male might be appropriate only if his height is significantly greater than to those of all other human males of the underlying domain.
- The necessity to incorporate more that domain knowledge *stricto-sensus* in the generator's ESE model. In early NLG systems, domain knowledge constituted the whole ESE model[36]. Later systems investigating lexical choice from the perspective of reference (KAMP [Appelt 85], EPICURE [Dale 88], FN [Reiter 90]) pointed out that knowledge of the hearer's domain knowledge should also constrain the wording of referring expression. For example, choosing *the giant* to refer to a very tall human male might be appropriate only if the hearer *knows* about his exceptional height. Note that the status of the hearer's encyclopedic knowledge or beliefs in our constraint classification (cf. section 2) is debatable. We chose to classify it as an encyclopedic constraint because it is typically represented in the same formalism as encyclopedic domain knowledge. It could, however, be arguably classified as an interpersonal constraint.
- The limitations of making *all* content determination decisions before any surface realization decision. For example, deciding how to word a definite description of a non-predicative concept token, a surface realization decision, can entail deciding what properties of the token to explicitly include in this description, a content determination decision. [Appelt 83], the first researcher to investigate generation of definite NPs from the perspective of reference, advocated the interleaving of content determination and surface realization. [Dale 88] argued that even the generation of clauses realizing predicative concept tokens is best considered as a problem of reference. He proposes a text generation model where content determination,

---

[35]This assymetry is partly rooted in the fact that nominals are constituents of the *simplest* clauses, whereas clauses are constituents of only quite *complex* NPs.

[36]Since these systems were not provided with any model of the ISE, the representation of the underlying domain was the only extra-linguistic knowledge they had. Furthermore, since their linguistic knowledge was typically represented procedurally in the generator's code, their domain knowledge was also the only declarative knowledge they had. This probably explains why what is generally intended by the *"knowledge base"* of a generator is not *all* the knowledge it is using but *only* the encyclopedic part of it. In this survey, we are using **Knowledge Base (KB)** in its broader sense, to refer to *all* the declarative knowledge used by a generator. We are using **Encyclopedic Knowledge Base (EKB)** to refer to the encyclopedic part of it, i.e. the ESE model of the generator.

discourse organization and surface realization are interleaved and recursively call each other.

- The necessity to integrate both discursive and encyclopedic factors to constrain lexical choices. For example, in a chain of successive references to a given token, all lexicalizations will be constrained by the same encyclopedic factors. However, they will considerably differ from each other because they appear at different point in the discourse structure[37].

In the next sections we survey a number of systems with emphasis on how they use encyclopedic constraints on lexical choice. We start with the systems that focus on the lexicalization of predicative concept by clauses. Among these systems, we first survey two systems resulting from lexical choice centered research efforts: BABEL [Goldman 75] and Danlos' generator [Danlos 86]. We then survey two systems resulting from application driven research efforts for which lexical choice was just one of the issues encountered to fulfill a practical generation need: PHRED [Jacobs 85] and ANA [Kukich 83]. We complete this section by presenting EPICURE [Dale 88], a text generator focusing on the lexicalization of non-predicative concepts by nominals.

---

[37]Most systems that we discuss in this chapter integrate both discursive and encyclopedic criteria to choose between alternative lexicalizations. We are therefore compelled to discuss some discursive constraints in this section devoted encyclopedic constraints.

5.2 BABEL

Historically, BABEL [Goldman 75] was the first generator to perform lexical choice. It generates single sentences from **Conceptual Dependency (CD)** networks. Its architecture is a pipeline of two modules. The first module is responsible for lexicalization and semantic grammaticalization. The second module is responsible for morpho-syntactic grammaticalization and linearization. The first module takes a CD network as input and produces a syntactic network of words and grammatical relations. The second module takes this syntactic network as input and produces an English sentence. The second module is implemented as an ATN (cf. [Woods 70] for the original definition of ATNs and [Simmons & Slocum 72] for their use for generation). In what follows we describe how the first module performs lexicalization of its CD input.

CD [Schank 75] was one of the first encyclopedic knowledge representation formalisms and it has been used by a wide variety of NLP programs. It was designed to be both language and domain independent. It is assymetric with regard to predicative and non-predicative concepts. It provides 15 primitives and 14 role relations for abstract predicative concepts[38]. However, it does not allow any abstraction of non-predicative concepts: they are represented as atoms. Predicative concept tokens are expressed as instances of CD primitives with CD role relations filled by other concept tokens. Different role values allow for the differentiation of the tokens of the same primitive. The CD representation of two sentences generated by BABEL is given in figure 5-1.

(1) *While drinking a beer, Steve smoked a cigarette.*

```
((concept-type INGEST)  ;;  CD primitive
 (agent STEVE)
 (object BEER)
 (through MOUTH)
 (time ((concept-type INGEST)  ;;  CD primitive
        (agent STEVE)
        (object CIGARETTE)
        (through MOUTH)
        (time PAST))))
```

(2) *Bill shot Steve, killing him.*

```
((concept-type STATE-CHANGE)  ;; CD primitive
 (object STEVE)
 (initial-state ALIVE)
 (final-state DEAD)
 (time PAST)
 (cause ((concept-type PROPEL)  ;; CD primitive
         (agent BILL)
         (object BULLET)
         (from-location GUN)
         (to-location STEVE))))
```

Figure 5-1: Conceptual Dependencies representations[39]

[38] This abstraction allows to factor out content elements shared by several lexical items as well as to represent implicit content.

[39] For the sake of clarity we paraphrased the rather cryptic CD icons by a standard attribute-value pairs notation similar to that of [Allen 87].

BABEL performs only one type of lexical choice: verb choice. For each predicative concept token in its input, BABEL chooses the verb of the clause realizing it. It constrains this choice by only one factor: the role values of the token to lexicalize. It does so by traversing a **discrimination network**. In such a network, the root is a CD primitive, the leaves are the verbs candidate for its lexicalization and the intermediary nodes are tests of the token role values. Given a token of some CD primitive, BABEL chooses a verb by finding a path to that verb in the discrimination net for that primitive. This path represents the semantic restrictions on the type of tokens the chosen verb accepts as role fillers. BABEL's discrimination net for the primitive INGEST is given in figure 5-2. To generate sentence (1) of figure 5-1 from the corresponding CD input, BABEL would traverse the INGEST net twice. The first traversal would give *drink2* and the second would give *smoke*.

Insert here fig.2 p.38 of Danlos' book

Figure 5-2: BABEL's discrimination net for INGEST (from [Danlos 86])

The nature of the tests on the INGEST discrimination net shows that, even for performing lexicalization of predicative concepts *only*, it is necessary to abstract non-predicative concepts. For example, the knowledge that CIGARETTE is a token of the OBJECT-TO-BE-SMOKED concept[40] is needed to choose *smoke* as the verb of the clause realizing the embedded token of INGEST in the input to sentence (1). Thus BABEL needs extraneous[41] information to perform verb choice: a property inheritance hierarchy of non-

[40]i.e. other tokens of this concept being, say, PIPE, CIGAR, JOINT etc.

[41]i.e. information not provided in its CD input.

predicative concepts. This hierarchy is BABEL's ESE model. In its input, BABEL interprets the atoms representing the non-predicative concept as pointers to this hierarchy.

The obvious restriction of the applicability of Goldman's work is that it does not attempt to address many important lexical choice issues:

1. *The mapping between content unit types and SLU types*. In BABEL, this mapping is one-to-one, as all predicative concept tokens are mapped onto clauses and all non-predicative concept tokens mapped onto NPs.

2. *The generation of attributive and equative clauses*. All the clauses generated by BABEL are action clauses. Attributive clauses are clauses such as *Steve has a white car*, equative clauses are clauses such as *Steve's car is white* and action clauses such as *Steve drives a white car*.

3. *The choice of other SLUs than verbs*. In BABEL the mapping between the other SLUs (e.g. NPs, adjectivals) and the token they realize is one-to-one.

4. *The choice of more or less specific verbs*. BABEL always chooses the most specific verb.

A less obvious restriction of the applicability of Goldman's work is that it provides a solution to a somewhat artificial problem: coarse and domain-independent verb choice. BABEL's verb choice is coarse because its vocabulary is very limited. Only a very few verbs are provided as potential lexicalizations for each CD primitive. For example, when the object of INGEST is a SOLID-SUBSTANCE, its only possible lexicalization is *eat*. A generator with a more extensive vocabulary would also include *graze, crunch, devour, swallow, gnaw, guzzle, chew, nibble, peck, gobble, wolfdown,* etc. A similar list could be provided for each leaf of BABEL's discrimination nets. As judiciously pointed out by [Danlos 86], a fine-grained choice between elements of such lists is a much more relevant issue than the coarse type of choice BABEL performs. The fine-grained choice represents alternative lexicalizations of a concept within a stereotyped domain of the real world, whereas the coarse type of choice represents alternative lexicalizations of a very abstract concept cutting across different domains of the real world (e.g. the concept EAT is a well-defined concept in the domain NUTRITION, whereas the CD primitive INGEST is a vaguely defined concept pervasive over almost all domains). But the state-of-the-art in both knowledge representation and computational linguistics makes the building of domain-independent NLG systems an intractable task. Thus, as a subtask of an overall intractable task, describing the alternative lexicalizations of very general concepts across real-world domains is of very little practical interest. To the contrary, describing the alternative lexicalizations of well-defined concepts in one restricted real-world domain is a task of great practical interest: NLG systems builders are continuously confronted with it. Thus, BABEL solves the artificial problem of choosing, from too limited a vocabulary, a verb to express too general a semantic primitive.

However, the technique it uses to solve this problem, discrimination nets, has been successfully used by several other generators developed after BABEL. In these generators, such as Danlos's generator [Danlos 86] and PAULINE [Hovy 87] reviewed later on in this survey, discrimination nets implement various types of constraints on lexical choice, not necessarily encyclopedic. One problem with discrimination nets is to find good criteria to determine whether a given test should appear above or below another test in the net. This problem is a particular manifestation of a general problem with taxonomizing formalisms, namely the difficulty of encoding the far from perfect natural taxonomies.

5.3 Danlos' generator

We introduced Danlos' work in section 3.1. In particular, we presented the generation algorithm of her system in section 3.1.3. One inherent characteristic of this algorithm, is that her system maps content unit types and SLU types in a one-to-one fashion: all predicative concept tokens are realized by clause patterns (in step 1 of the algorithm) and all non-predicative concept tokens are realized by nominals (in step 5 of the algorithm). Both these realizations are constrained by encyclopedic factors. This is why they are reviewed in this section. The realization of predicative concepts by clause patterns is presented first, followed by that of non-predicative concepts by nominals. Danlos' implementation of the constraining process between lexical and grammatical choices was discussed in section 3.1.4.

5.3.1 Choice of clause patterns

Recall that Danlos' systems generates a 1 or 2 sentence long report from a frame-based representation of a terrorist attack. It chooses one clause pattern for each predicative concept token in this representation. Each of these choices is performed by traversing a discrimination net. The representation of a terrorist attack contains two top-level tokens: one token of the concept ATTACK-ACTION, representing the attacking action and one token of the concept ATTACK-RESULT, representing the result of the attack. The ESE model of Danlos' generator consists of two inheritance hierarchies: one for predicative concepts and one for non-predicative concepts. ATTACK-ACTION and ATTACK-RESULT are located at the first level of the predicative concepts hierarchy. Other concepts (e.g. VICTIM-LOCOMOTION) whose tokens can be role fillers of an ATTACK-ACTION token are also located at that level. There is one discrimination net associated with each concept of that level. The discrimination net for ATTACK-RESULT is given in figure 5-3.

The common characteristics of the respective discrimination nets used by Goldman and Danlos are the following:

- They both implement the lexicalization of predicative concept tokens.
- The tests on their intermediary nodes implement encyclopedic constraints *only* (no tests implement any discursive or interpersonal constraints).
- Although the leaves of BABEL's nets are single words (verbs) and those of Danlos' generator are phrasal patterns (clause patterns), there is not much of a difference between the two. In Danlos' generator, except for the verb phrase and some related closed-class items (preposition, particle etc), all clause pattern constituents are variables. Therefore, as far as lexical choice is concerned, little more than verb choice directly stems from clause pattern choice in Danlos' generator[42].

The differences between the respective use of discrimination nets by Goldman and Danlos are the following:

- In BABEL, the tests encountered during the lexicalization of a token tests only the information contained in that token: the primitive it is an instance of, its role structure and its role values. In Danlos' generator, they test any part of the overall input description the token is part of. They thus test a larger part of the encyclopedic context.

---

[42]However, some *grammatical* choice, such as voice choice, also directly stems from clause pattern choice

Insert here figure 10 p.144 of Danlos book

Figure 5-3: Discrimination net to lexicalize the concept ATTACK-RESULT (from [Danlos 86])

- In BABEL, the roots of the nets are very general *domain-independent* concept (the CD primitives). In Danlos' generator, they are the specific concepts *of a very restricted encyclopedic domain*. Danlos thus uses discrimination nets to implement the fine-grained type of lexical choice discussed in section 5.1.

5.3.2 Microcoding of NPs and pronominalization

Danlos' generator realizes non-predicative concept tokens by nominals, i.e. by either NPs or pronouns. Although it microcodes these NPs, Danlos' generator does *not* have a declarative word-based lexicon. The words it uses, together with their encyclopedic and grammatical properties, are scattered in the code of the LISP functions called for the realization of non-predicative concept tokens. The NPs constructed by these functions, may contain 4 types of open-class SLUs: head nouns, pre-modifying adjectivals[43], restrictive relative clauses and non-restrictive relative clauses. The mapping between these 4 SLU types and the content unit types (in this case the types of the token slots) is *static*: Danlos' simply divided *at knowledge acquisition time* the various non-predicative concept slots of her restricted domain into the 6 types. Each of

[43]including participles.

these slot types is defined in terms of the types of SLUs that can be used for their realization:

- Type-1 slots are realized by head nouns.
- Type-2 slots are realized by adjectivals.
- Type-3 slots are realized by restrictive relative clauses.
- Type-4 slots are realized by non-restrictive relative clauses.
- Type-5 slots are realized by additional sentences.
- Type-6 slots are realized by grammatical properties of the nominal.

Type-6 slots such as NUMBER or DEFINITE are not lexicalized but grammaticalized. We will therefore not discuss them any further. Figure 5-4 gives 4 examples of non-predicative concept tokens with the corresponding types of their slots and a text containing a realization of them.

```
  tokens                 slot-types       tokens                slot-type

((token person-1)                       ((token person-2)
 (isa PERSON)                            (isa PERSON)
 (first-name Steve)         1            (first-name Karen)         1
 (profession student)       1            (profession student)       1
 (particularity drunk)      3            (particularity drunk)      4
 (age 46)                   2            (definite yes)             6
 (definite yes)             6            (number 1))                6
 (number 1))                6

Steve, who was drunk, mumbled           The student who was drunk mumbled


((token seat-1)                         ((token bomb-1)
 (isa SEAT)                              (isa BOMB)
 (type chair)               1            (type remote-controlled)   2
 (color red)                2            (charge 4kg)               5
 (definite no)              6            (in-location sewers))      5
 (number 1))                6

The red chair                           Anarchists exploded a remote-controlled
                                        bomb. The bomb, which was hidden in the
                                        sewers, contained 4 kilos of dynamite.
```

Figure 5-4: Examples of non-predicative concept tokens in Danlos' generator

The examples of figure 5-4 show that the Danlos' slot typing is concept-dependent: the slot *type* is of type 2 in a token of the concept BOMB but of type 1 in a token of the concept SEAT. It also shows that Danlos' generator mapping between content unit types and SLU types at the NP level is not only static, but ad-hoc. It is not based on ontological principles of any generality, but on concept-dependent heuristics. These concept-dependent heuristics incorporate some stylistic constraints (e.g. the choice of mapping the CHARGE and IN-LOCATION of a BOMB token onto additional sentences is motivated by the stylistic constraint of avoiding the generation of sentences containing too many adjectivals and relative clauses.). Danlos' generator is guided by these heuristics to decide which of the type 2 to type 6 slots to realize and by which types of SLUs. Once it makes this decision, it maps the slot value onto words in a one-to-one fashion. It therefore not only avoids the issue of dynamically[44] choosing the modifier type, but also that of choosing inside a given type, the most appropriate modifier to lexicalize the slot value.

The heuristics Danlos' generator uses for head noun choice and pronominalization are given figure 5-5. Since they depend on whether a token occurrence is initial or not, they integrate both *discursive* and encyclopedic constraints. They have two major limitations:

- Although they consider several slots from the input token to be realized by the head noun, once a particular slot has been chosen, they do not considers alternative lexical items for each value of that slot.
- They handle pronominalization simplistically. Aside from causing some cases of over-pronominalization and under-pronominalization, their applicability is restricted to texts containing at most 3 clauses, like those produced by Danlos' generator.

For the first occurrence of a non-predicative concept token **T**:

(h1): *IF* **T** has no type-1 slot *THEN* generate an NP and chooses the value of the ISA slot as the head.

(h2): *IF* **T** has exactly one type-1 slot *THEN* generate an NP and chooses the value of that type-1 slot as the head.

(h3): *IF* **T** has several type-1 slot *THEN* generate an NP and choose the value of one of these type-1 slots as the head[45].

For subsequent occurrences of **T**:

(h4): *IF* all the type-1 slots of **T** have already been used in preceding occurrences of **T** *THEN* generate a pronoun.

(h5): *IF* exactly one type-1 slots of **T** have not yet been used in preceding occurrences of **T** *THEN* generate an NP and chooses the value of that remaining type-1 slot as the head.

(h6): *IF* several type-1 slots of **T** have not yet been used in preceding occurrences of **T** *THEN* generate an NP and chooses the value of one of these remaining type-1 slots as the head[46].

Figure 5-5: Head noun lexicalization and pronominalization heuristics in Danlos' generator

Danlos' generator NP microcoding is further evaluated by comparison with those of other systems later on in this survey. First from the perspective of initial reference in section 6.4.1 of the present chapter, then from the perspective of subsequent references in section 5 of chapter 7.

---

[44]i.e. at generation time.

[45]The choice between these several type-1 slot values is concept-dependent.

[46]In this case too, the choice between these several type-1 slot values is concept-dependent.

5.4 PHRED

5.4.1 An overview of PHRED

PHRED (PHRasal English Diction) [Jacobs 85] is the surface generation component of the **Unix Consultant (UC)** [Wilensky & al. 84]. UC is an online help system interacting in natural language with UNIX users. PHRED generates single sentence responses to questions about UNIX commands. In contrast to the generators we reviewed in the two preceding sections (BABEL and Danlos' generator), PHRED does not result from a research effort focused on lexical choice. Motivated by its integration within an interactive help system, the issues emphasized for PHRED 's development were:

- Knowledge structure sharing between PHRED and the question understanding component of UC, **PHRAN (PHRasal ANalyzer)**,
- Speed of response generation.

[Danlos 86] and [Danlos, Guez & Sabbagh 85] present INTERIX, an interactive system implementing the same functionality as UC but based on a different design. In particular, INTERIX's response generator is based on Danlos' generation model. Danlos' model fulfills one of PHRED requirements: fast response generation. However, all its power is not required for such an application and it does not share any knowledge structure with INTERIX's question understanding component. We present PHRED as an example of generator addressing the issue of lexical choice in a practical way, without making it the central issue of its design.

In UC all content determination is done by the response planner before PHRED is called upon. The input of PHRED consists in a concept token to realize together with a set of linguistic constraints on its realization. Figure 5-6 gives an example of such input and the sentence PHRED generates from it. In UC, concepts are represented by arbitrary deep lists, containing variables. Concept tokens are represented the same way, except that they do not contain any variables. The more general a concept, the more variables its representation contains. PHRED's ESE model consists of a set of such representations of UNIX concepts.

```
Constraints: POS=sentence
             voice=active

Token: (planfor
        (result
         (state-change
          (object file1)
          (state-name location)
          (from (inside-of (object current-directory)))
          (to (not (concept (inside-of (object current-directory)))))))
        (method (mtrans (actor *user*)
                        (object (command (name rm)
                                         (args (filename))))
                        (from *user*)
                        (to *UNIX*))))
```

Typing "rm filename" causes the file filename to be removed from the current directory.

Figure 5-6: An example of PHRED's input and the corresponding output

To carry out generation, PHRED uses declarative knowledge structures called **Pattern-Concept Pairs (PCPs)**. There are two types of PCPs: **Conceptual PCPs** and **Ordering PCPs**. Conceptual PCPs express constraints on the mapping of content units onto SLUs. They are made of three components: a *concept*, a *pattern* and a set of *properties*.

- The *concept* is a UNIX concept represented as described just above.
- The *pattern* specifies conjunctions and/or disjunctions of constraints on the SLUs that can be used to realize the *concept* tokens. It consists of an ordered set of constrained constituents. The constraints, on each constituent, can be either grammatical and/or lexical and/or encyclopedic.
- The *properties* express relations between *concept* elements and *pattern* constituents.

An example of conceptual PCP is given in figure 5-7. Ordering PCPs are made of two components: a *pattern* and a set of *properties*. They express grammatical constraints on linearization (e.g. the constraint on constituent ordering for passive sentence).

```
Pattern: <agent> <root=remove> <physobj> <<word=from> <container>>

Concept: (state-change (object ?rem-object)
                           (state-name location)
                           (from (inside-of (object ?cont)))
                           (to (not (concept (inside-of (object ?cont))))))

Properties: tense = (value 2 tense)
            forms = (active-sentence passive-sentence)
            rem-object = (value 3)
            cont = (value 5)
```

- The pattern specifies the constraints on each constituent of the clause pattern built around the verb *to remove*.
- The concept specifies which token can be realized by that clause pattern[47].
- The 1st property specifies that the tense of the whole clause is the tense of the verb.
- The 2nd property specifies that the clause pattern can be both used at the active and passive form.
- The 3rd property specifies that the value of the ?rem-object variable of the concept fills the 3rd constituent of the clause pattern.
- The 4th property specifies that the value of the ?cont variable of the concept fills the 5th constituent of the clause pattern.

Figure 5-7: Conceptual PCP for clauses built around the verb *to remove*

In PHRED, generation proceeds recursively through a pipeline of three phases: **fetching**, **restriction** and **interpretation**. For each constituent of a given level[48], PHRED repeatedly goes through this pipeline until the realization of that constituent is completed, i.e. until its pattern is fully instantiated

During the fetching phase, a hashing-scheme over the PCP-base is used to retrieve:

1. A list of conceptual PCPs whose concept *partially* matches the input token. This list is

[47]This is PHRED's domain-dependent semantics for the verb *to remove*.

[48]The top-level constituent being the sentence and the bottom-level the individual word.

ordered by degree of concept specialization.

2. An ordering PCP whose properties matches the input linguistic constraints.

The restriction phase is decomposed into five steps:

1. Scanning of the ordered list of fetched conceptual PCPs for an element **C** whose concept *completely* matches the input token.
2. Instantiation of the variables of the concept of **C**.
3. Instantiation of the corresponding variables of the pattern of **C** (recall that the correspondence is specified in the properties of **C**).
4. Combination of the properties of **C** with the properties of the fetched ordering pattern **O**. If it fails, go back to step 1.
5. Combination of the partially instantiated pattern of **C** with the pattern of **O**. If it fails, go back to step 1.

The interpretation phase consists in the application of the property combination constructed in step 4 of the restriction phase to the pattern combination constructed in step 5 of the restriction phase. If it succeeds the whole pipeline is recursively called on the yet uninstantiated elments of the pattern. If it fails, PHRED goes back to step 1 of the restriction phase.

5.4.2 Lexical choice in PHRED

Each time a fetching-restriction-interpretation sequence completes, it results in a set of various surface realization choices, possibly including lexical choices. Let us consider for example, the realization of the sentence of figure 5-6. The choice of the verb *to remove* occurs during the call to the fetching-restriction-interpretation pipeline on the infinitive-clause constituent of the top-level clause pattern. The PCP of figure 5-7 is chosen because: (1) it is the first fetched and (2) both its restriction and its interpretation succeed.

PHRED performs surface realization is very similar to functional unification, except for the hashing-scheme of PHRED’s fetching phase. This scheme allows a significant reduction of the backtracking involved with unrestricted unification[49]. Although many generators uses functional unification to carry out surface realization, none of them implements all four aspects of it by *a single* unification process: [McKeown 85b] and [Appelt 83] do *not* use functional unification for lexicalization and semantic grammaticalization, while [McKeown & al. 90] use a pipeline of *two separate* unification processes[50], one for lexicalization and semantic grammaticalization and the other for morpho-syntactic grammaticalization and linearization. Thus, with respect to the way it performs lexical choice, PHRED differs from both BABEL and Danlos’ generator, which have dedicated processes to perform lexical choice. Using a single process, like in PHRED to perform simultaneously all types of surface realization choices allows any lexical choice to occur before or after any other surface realization choice. It also allows any lexical choice to both constrain and be constrained by any other surface realization choices.

Let us now compare PHRED to both BABEL and Danlos’ generator with respect to other characteristics:

---

[49]Alternation indexing supported by functional unifiers such as FUF [Elhadad 88], also allows reduction of backtracking.

[50]i.e. no backtracking is allowed between the two.

- *Mapping between content types and SLUs*: in BABEL and Danlos' generator, this choice is necessarily one-to-one because different techniques are used for choosing different types of SLUs. In PHRED, a many-to-many mapping could be implemented without difficulty using PCPs.
- *Types of SLUs for which authentic choice is implemented*: like BABEL and Danlos' generator, the only SLUs PHRED *actually* chooses at generation time are verbs[51]. However, any type of SLU choice could be implemented using PCPs.
- *Part of the encyclopedic context taken into account*: like BABEL and unlike Danlos' generator, PHRED constrains the lexicalization of a concept token exclusively by the information contained in that token.
- *Granularity of the lexical choice*: like Danlos' generator and unlike BABEL, PHRED performs fine-grained lexical choice in a restricted domain. Although the most general concepts of PHRED's ESE model are CD primitives, the most specific ones are UNIX entities. PHRED thus uses CD primitives more like the Upper-Model of PENMAN (cf. section 3.2.) than like BABEL.

Finally, let us note that PCPs are extremely heterogeneous knowledge structures integrating many different types of knowledge. This heterogeneity has the advantage of making the interaction of different types of constraints easy to implement. It also has the major drawback of precluding any use of available compiled sources of linguistic knowledge potentially portable across encyclopedic domains (such as grammar rules, idiosyncratic grammatical properties of lexical items and collocations).

[51]or microcoded clause patterns.

5.5 ANA

5.5.1 An overview of ANA

[Kukich 83] presents **ANA**, a system generating natural language reports from an online stock market database. A report generated by ANA summarizes, in three paragraphs of one or two sentences, the daily fluctuations of several stock market indices. Half-hourly updates of these indices values constitute ANA's input. An example of input is given in figure 5-8. The corresponding report generated by ANA is given in figure 5-9.

Insert here excerpts from figure 4.0.1.1. of [Kukich 83] p.36

Figure 5-8: An input set of ANA (from [Kukich 83])

Among the generators discussed in this section, ANA is remarkable in that it is an attempt to fulfill a very practical generation need[52]. Moreover, it distinguishes itself drastically by taking as input *numerical data*, as opposed to generation-oriented representations of the underlying domain. As pointed out by [McKeown & Swartout 87] among others, domain knowledge representations developed for non-generation purposes, generally constitute an inappropriate input for NLG systems. Thus, besides content

[52]Although PHRED also addressed a practical problem, the motivation underlying the development of other generators discussed in this section (BABEL, Danlos' generator, and EPICURE) was essentially theoretical.

Thursday June 24, 1982

wall street's securities markets meandered upward through most of the morning, before being pushed downhill late in the day yesterday. the stock market closed out the day with a small loss and turned in a mixed showing in moderate trading.

the Dow Jones average of 30 industrials declined slightly, finishing the day at 810.41, off 2.76 points. the transportation and utility indicators edged higher.

volume on the big board was 558600000 shares compared with 62710000 shares on Wednesday. advances were ahead by about 8 to 7 at the final bell.

Figure 5-9: The report generated by ANA from the data of figure 5-8

determination, discourse organization and surface realization, report generators like ANA have to carry out an additional generation task: *data abstraction*. It consists of abstracting, from the raw numerical data, the underlying domain concept tokens to talk about in the report.

ANA's architecture is a pipeline of four modules: a **fact generator**, a **message generator**, a **discourse organizer** and a **text generator**. The first is written is C, the remaining three in OPS5.

The fact generator takes as input the raw numerical data from the online stock market database, and compile them into **facts**. Facts are aggregate of various statistics over a given period of time. They are represented by OPS5 **Working Memory Elements (WMEs**). An example of fact is given in figure 5-10.

The message generator takes facts as input and output **messages**. Abstracted from facts, messages are clause-sized content units. They correspond to the concept tokens of generators taking symbolic inputs. They are also represented by OPS5 WMEs. An example of message (with the corresponding clause generated from it by ANA) is given in figure 5-10. Each message is an instance of a given message type, and message types are hierarchically organized. ANA's hierarchy of messages constitutes its ESE model.

The discourse organizer takes as input the unstructured set of messages produced by the message generator. It groups them into paragraphs and orders them inside these paragraphs. It also collapses some messages together (e.g. when two consecutive messages differ exclusively with respect to their subjects, they are collapsed into one message with a compound subject[53]).

The text generator takes as input the ordered set of messages produced by the discourse organizer. It maps each of these messages onto a clause. It then combines the clauses together to form the English text of the report.

Thus, the partition of generation tasks among these four modules is the following:

- *Data abstraction* is carried out in part by the fact generator and in part by the message generator.
- *Content determination* is carried out as a side-effect of data abstraction. It is thus also carried out in part by the fact generator and in part by the message generator.

[53]The text generator subsequently realizes such a compound subject by a conjunctive nominal.

- *Discourse organization* is carried out by the discourse organizer
- *Surface realization* is carried out by the text generator.

Example of fact:

```
(make fact
    ^fact-name half-hour-stat
    ^indicator-name Dow-Jones-industrial-average
    ^indicator-type composite
    ^date 04/21
    ^time 11:30am
     ^current-level 1192.82
    ^direction down
    ^degree 2.03
    ^cumulative-direction up
    ^cumulative-degree 1.35
    ^high-or-low-mark nil)
```

Example of message:

```
(make message
    ^date 6/24
    ^topic Dow-Jones-industrial-average
    ^subtopic Dow-Jones-industrial-average-status
    ^subject-class Dow-Jones-industrial-average
    ^direction down
    ^degree small
    ^time closing
    ^variable-level 810.41
    ^variable-degree 2.76)
```

Corresponding sentence: *"the Dow Jones average of 30 industrials declined slightly, finishing the day at 810.41, off 2.76 points"*.

Figure 5-10: Intermediary internal representations in ANA

Now that we have seen the overall architecture of ANA, let us focus on surface realization. It is both *linear* and *macrocoded*: linear, because the text generator takes as input an ordered list of messages and processes them one after the other; macrocoded because each message is mapped onto clause patterns that are *stored* in a phrasal lexicon. These clause patterns contain two types of variables: one subject variable and zero, one or several numerical variables. The subject variable is instantiated by one of the nominals hard-wired in the phrasal lexicon. The numerical variables are instantiated by the values of the corresponding message numerical attributes. Aside from the subject and the numerical values, all the other constituents (e.g. complements, adjuncts) of the clause patterns are hard-wired. All the clause constituents realizing the predicate function of the clause are thus retrieved as a single unit in the phrasal lexicon. Therefore, ANA's mapping of one message onto one clause basically consists of retrieving from the phrasal lexicon an appropriate clause pattern, instantiating it and combining it with the clause patterns chosen for the adjacent messages. More precisely, ANA's overall surface generation algorithm is the following[54]:

[54]Some minor steps involving much simplified choices of seldomly used SLU types have been omitted here for simplicity's sake.

1. Choice in the phrasal lexicon of a clause pattern to lexicalize the semantic predicate of the message.
2. Instantiation of the numerical variables of this clause pattern with the values of the corresponding message numerical attributes.
3. Choice of the syntactic form of the clause.
4. If the chosen syntactic form requires a connective with the preceding clause (i.e. the clause lexicalizing the immediately preceding message), choice of the appropriate connective.
5. Choice in the phrasal lexicon of a nominal to lexicalize the semantic subject of the message.
6. Choice of the verbal inflection enforcing subject-verb agreement.
7. Spelling out of the linguistic string resulting from the choices made in steps 1-6.

Special WMEs called **medial text elements** are used to hold temporary information from one step to another. These medial text elements are heterogeneous knowledge structures integrating encyclopedic, discursive, lexical and grammatical information. However, because each step of the surface realization algorithm is carried out by firing a specialized set of OPS5 productions, lexicalization (performed at steps 1,4 and 5) and grammaticalization (performed at steps 3, 6 and 7) are carried out as two *interleaved* and *modular* processes. In this respect, ANA is closer to Danlos' generator than to PHRED, in which lexicalization and grammaticalization are integrated into a single process. With respect to the mapping between content unit types and SLU types, ANA performs a one-to-one mapping like all the other generator we surveyed in this section. However, unlike BABEL and Danlos' generator it does not so because it uses different techniques to choose different SLU types, but mainly for the two following reasons:

- The control information contained in the goals of the OPS5 productions used for surface realization. These productions could be easily modified to allow a many-to-many mapping.
- The high level of macrocoding of the entries of its lexicon. Especially, since the only variable constituent in ANA's clause pattern is the subject, only this constituent could be realized by other SLU types than nominals. For the others, since there no mapping at all the question of a many-to-many mapping does not even arise, and subject is hardly the constituent for which the availability of a variety of SLU types is the most needed. Thus, having a many-to-many mapping between content unit types and SLU types in an ANA-like generator would not result in a dramatic increase in expressive flexibility without a finer microcoding of the dictionary entries.

5.5.2 Lexical choice in ANA

5.5.2.1 Choice of clause pattern

The choice of the clause pattern lexicalizing the semantic predicate of a message is the first step of ANA's overall surface realization algorithm. It is triggered by the firing of a dedicated set of OPS5 productions, and essentially performed by matching the message's predicate encyclopedic attributes onto the corresponding attributes in the clause pattern entries of the phrasal lexicon. Each of these entries is represented by an OPS5 WME specifying a mapping between a domain concept and a clause pattern candidate to its lexicalization. An example of such an entry in given in figure 5-11. In this entry, the first nine attributes are encyclopedic. They define a predicative domain concept: the class of messages with matching encyclopedic attributes. The following five attributes of the entry are linguistic. They define a clause pattern. The remaining two attributes are discursive. They are used to choose between several entries whose semantic attributes match those of the message to lexicalize.

```
(make phraselex
    ^phrase-type predicate
    ^topic general-market
    ^subtopic interesting-market-fluctuation
    ^duration long
    ^degree small
    ^direction up
    ^time early
    ^subject-type name
    ^subject-class market
    ^verb-past-singular-inflection meandered
    ^verb-past-plural-inflection meandered
    ^verb-present-participial-inflection meandering
    ^verb-infinitive-inflection to meander
    ^predicate-remainder upward through most of the morning
    ^random 5
    ^length 14)
```

Figure 5-11: Clause pattern example of a predicate entry of ANA's lexicon

Overall, the clause pattern choice procedure is the following:

1. Retrieve the entries whose encyclopedic attributes match those of the predicate part of the message to lexicalize.
2. If several entries are retrieved, break the tie using the *length* attribute (containing the approximate length of the corresponding clause pattern in syllabus) and the heuristic that two long sentences followed by a short one is stylistically appropriate in stock market reports.
3. If several entries are still candidate, generate a random number between 1 and 10 and pick-up the entry with the closest *random* attribute.

Let us now compare ANA's lexicalization of predicative concepts by clauses to that of the other generators surveyed in this chapter with respect to the following characteristics:

- *Part of the encyclopedic context taken into account*: like BABEL and PHRED but unlike Danlos' generator, the sole encyclopedic constraint on the choice of a clause realizing a token is the information contained in the token itself. In the case of ANA the token is represented by a message. Although the lexicalization of a given message in ANA takes into account grammatical and discursive constraints coming from previous messages (stored in the medial text elements) it does not take into account the encyclopedic attributes of these previous messages.
- *Granularity of the lexical choice*: like Danlos' generator and PHRED but unlike BABEL, ANA performs fine-grained lexical choice in a very restricted domain. As a matter of fact, the encyclopedic knowledge needed to generate stock market reports is probably significantly simpler to represent than that needed to generate terrorist attack reports or explanations about UNIX commands.
- *Surface linguistic coverage*: ANA covers a much wider variety of clauses than any other

existing complete text generation system[55]. In terms of structural types, ANA can use finite, present participial and past participial clause. In terms of function, ANA uses clauses as simple sentence, coordinated clause, adjunct subordinate clause and adverbial subordinate clause.

- *Level of clause microcoding*: ANA's clause pattern are more macrocoded than those used by both Danlos' generator and PHRED. ANA's macrocoding approach implicitly provides the ability to implement a many-to-many mapping between message semantic attributes and clause constituents. Figure 5-12 shows three clauses with three radically different structures used to convey the same combination of semantic attributes.

```
encyclopedic attributes of a message: ^ direction up
                                      ^ degree great
                                      ^ time early
```

3 different types of clause structures to realize them:
- *opened strongly*
- *picked up momentum early in trading*
- *got off to a strong start*

Figure 5-12: Many-to-many mapping between semantic attributes and clause constituents in ANA (from [Kukich 83])

5.5.2.2 Choice of nominals

The choice of the nominal lexicalizing the semantic subject of a message is the fifth step of ANA's overall surface realization algorithm. This nominal can be either:

- A descriptive NP: either a definite description (e.g. *the stock market*) or a determinerless description functioning like a proper name (e.g. *stock prices*).
- Neutral personal pronoun.
- The empty string (i.e. ellipsis of the subject as a whole).

The nominals ANA uses are entirely hard-wired in its phrasal lexicon. An example of such an entry is given in figure 5-13. The first four attributes are encyclopedic. They define a non-predicative domain concept: the class of messages with matching encyclopedic attributes. The next two attributes are linguistic. They define a nominal. The remaining three attributes represent discursive constraints. The hyponym-level attribute determines the degree of specificity of the entry as a referring expression. It is used to avoid repetition in wording successive references to the same token or to different tokens of the same topic. It is used in combination with the heuristic stating that successive references need to be less and less specific. The usage attribute is also used to avoid repetition. Everytime the entry is used in a given text, the value of this attribute is incremented and when several entries share the same encyclopedic and hyponym-level attributes, the one with the lowest usage attribute is chosen. Finally, the random attribute is used, as in clausal entries, as the final tie-breaker between otherwise undistinguishable entries. Appropriate NP entries are retrieved from the phrasal lexicon by matching their encyclopedic and discursive attributes against

[55] Let us recall here that we call a *complete* text generation system, a system carrying out in a non-straightforward way all three text generation subtasks: content determination, discourse organization and surface realization. It thus excludes stand alone grammatical components such as NIGEL [Mann & Matthiessen 83] or MUMBLE [McDonald 80].

those of the message to lexicalize.

```
(make phraselex
    ^phrase-type subject
    ^topic general-market
    ^subject-type name
    ^subject-class market
    ^subject-string wall street's securities markets
    ^subject-number plural
    ^hyponym-level 1
    ^random 4
    ^usage 0)
```

Figure 5-13: NP example of subject entry in ANA

The overall nominal choice algorithm of ANA is presented in figure 5-14. In the very simple domain of ANA where all the objects are financial indexes, macrocoding the generation of nominals referring to these objects is adequate. However in domains where the objects are significantly more complex and varied, such as the cooking domain of EPICURE [Dale 88] that we survey in the next section, this approach is no longer economical[56], even for achieving the minimal variety of referring expressions required to avoid generating too awkward looking texts. ANA's approach of nominal choice is compared to other approaches in section 7.5.

[56]i.e. it would require too much hand-coding of phrasal patterns at lexicon building time.

```
1/ Retrieve from the phrasal lexicon the NP entries whose encyclopedic
   attributes match those of the incoming message. The linguistic attributes
   of these entries specify the NPs candidate for the lexicalization of the
   token defined by the encyclopedic attributes of the message (i.e. the
   phrase-type, topic, subject-class and subject-type attributes)

2/ If the previous discourse does not contain any reference to a token
   sharing the same topic attribute then:
   - Select among the candidate entries those with hyponym-level = 0.
   - Among them, choose the entry with minimal usage and closest random
     attribute.

3/ If the previous discourse contains N reference(s) to different
   token(s) sharing the same topic attribute, then:
     - Select among the candidate NP entries those with hyponym-level = N.
     - Among them, choose the entry with minimal usage and closest random
       attribute.

4/ If the previous discourse contains N reference(s) to the same token then:
   - If the grammatical context allows for ellipsis[57], then:
     - "Lexicalize" the token by the empty string (i.e. ellide the subject)
   - If the grammatical context allows for pronominalization[58], then:
     - Use a neutral personal pronoun
     else:
     - Select among the candidate NP entries those with hyponym-level = N.
     - Among them, choose the entry with minimal usage and closest random
       attribute.
```

Figure 5-14: ANA's nominal choice algorithm

5.6 EPICURE

[Dale 88] presents EPICURE, a system generating cooking recipes made of two parts:

- *An ingredient list*: nominals referring to the ingredients of the recipe.
- *A cooking action list*: simple clauses specifying the successive actions to be performed by the reader[59] of the recipe in order to prepare a given meal.

An example of a cooking recipe generated by EPICURE is given in figure 5-15.

Cream of Butter Bean Soup

four ounces of butter beans
a large onion
a medium potato
two carrots
two sticks of celery
one ounce of butter
1.5 pints of unsalted stock
0.5 pints of milk
four tablespoons of cream
some sea salt
some fresh ground black pepper
some grated nutmeg

Soak, drain and rinse the butter beans. Peel and chop the onion. Peel and chop the potato. Scrape and chop the carrots. Slice the celery. Melt the butter. Add the vegetables. Saute them. Add the butter beans, the stock and the milk. Simmer. Liquidise the soup. Stir the cream. Add the seasonings. Reheat.

Figure 5-15: An example of cooking recipe generated by EPICURE

As input EPICURE takes:

- An ordered list of semantic representations, one for each ingredient.
- The name of the meal whose recipe is to be generated.

First, EPICURE linearly scans the input list of ingredient representations and generates one referring NP for each element in this list. Then, EPICURE constructs a hierarchical text plan for the recipe from hand-coded plan skeletons stored in a plan library. Finally, EPICURE realizes this text plan by generating a simple clause for each atomic cooking action contained in the plan.

The particular lexical choice issue addressed by EPICURE is the wording of *nominals* in chains of *successive references* to *physical objects* of *various internal structures* in a *dynamically changing* domain. This is in sharp contrast with the other generators that we have surveyed in this section which address the issue of wording *clauses* in a *static* domain where *all objects* have a simple *individual* structure. The

[59]In the remaining of the section we will refer to the user of the recipe as *the hearer*.

specificity of the lexical choice issue investigated by Dale led him to make two important contributions. The first of these contribution lies in the originality of EPICURE's architecture, which is presented in figure 5-16. EPICURE is original both in terms of the knowledge sources available to its different processing modules and the control flow between these modules. The second of these contributions is a formalism for the semantic representation of physical objects of diverse and potentially evolving internal structures. Both of these contributions are surveyed in the next section where we overview EPICURE. In the subsequent section, we elaborate on EPICURE's surface realization process during which lexical choice is carried out.

5.6.1 An overview of EPICURE

The top-level distinction in Dale's ontology of cooking domain entities is between **Eventualities** and **Generalized Physical Objects (GPOs)**. It corresponds to the distinction between predicative and non-predicative concepts used throughout the present survey. The term *Eventuality* comes from the general ontology of predicative concepts of [Bach 86] which groups diverse kinds of states, events, processes etc. In EPICURE, however, there is only one category of Eventuality: the cooking actions one performs when following a recipe. Depending on the level of detail of the recipe, some of these Eventualities are atomic and others are decomposed into lower-level ones. Atomic Eventualities are all realized by simple clauses like: *slice the celery*. The recipe as a whole is the highest-level Eventuality. Both Eventualities and Generalized Physical Objects are represented by **Features Structures (FSs)**. An feature structure is a set of attribute-value pairs, with each value being either an atomic symbol or recursively an feature structure[60]. Feature structures are thus very much like Functional Descriptions[61] ([Kay 79], [Elhadad 88]) and in particular they also allow for partial specifications gradually enriched through functional unification.

EPICURE's input is a feature structure containing a single feature **substance** specifying the meal to prepare using the recipe to generate. For example, the input corresponding to the recipe of figure 5-15 is *((substance bean-preparing))*. The architecture of EPICURE is described in figure 5-16. The input substance constitutes the top-level goal of the **discourse planner**, whose task is to build a hierarchical text plan for the recipe. To do so it uses three knowledge sources:

- The **plan library**: In this library, plans are represented by feature structures containing the following features: *substance*, *preconditions*, *effects* and *constituents*. The substance feature specifies the goal the plan satisfies. The constituents specifies the substeps in which it can be expanded. The top-level plans are recipe skeletons. The lower-level plans are decompositions of the constituents of higher-level plans either in terms of atomic actions or recursively in terms of lower-level plans.

- The **hearer model**: It contains the list of cooking actions the hearer knows how to carry out. It allows the discourse planner to decide whether or not to expand a given action into its substeps in the recipe plan (e.g. If the hearer knows how to chop mushrooms, there is no need to include a description of how to do it in the recipe. The simple request *chop the mushrooms* will do). It thus defines action atomicity.

- The **world model**: The main peculiarity of cooking recipes compared to the texts produced by other existing NLG systems is that from one point in the recipe to another, domain entities may considerably change (e.g. after the request *chop the onion* the onion is no longer a whole object). Some of them may also disappear and new ones may be created (e.g. the request

[60]It can also be either a list of atomic symbols or a list of feature structures.

[61]except for the special grammatical features of functional descriptions, CAT and CSET.

separate the yolk from the white causes the disappearance of an egg and the creation of both a white of egg and a yolk of egg). Therefore, generating appropriate referring expressions to these domain entities requires maintaining a record of the currently existing ones together with their current state. In EPICURE this record is the world model. It is a set of feature structures, one for each current entity of the underlying domain.

Insert here figure 4.1 p.145 of Dale’s thesis

Figure 5-16: The architecture of EPICURE (from [Dale 88])

The discourse planner unifies its input with the plan library, retrieving a plan whose substance feature matches that of the input. The preconditions of the plan are then unified with the world model. Upon their satisfaction, the input is enriched by the constituents feature of the plan, and the effects of the plan are passed to the **domain modeller** whose task is to accordingly update the world model. The hearer model is then consulted for each constituent feature of the enriched input, to determine which of them are atomic from the hearer’s knowledge perspective. Atomic constituents can be requested to the hearer without further expansion. Each non-atomic constituent recursively becomes a goal to unify with the plan library.

The output of the discourse planner, called the **discourse specification** is a completely expanded plan for the input goal. This plan is made of two parts:

- A list of Generalized Physical Objects to describe in the ingredient list,
- A tree of Eventualities whose structure reflects the hierarchical decomposition of the recipe. The leaves of that tree are the atomic cooking actions to request in the recipe.

This discourse specification is passed to the **discourse generator** which interacts with two other components of the system:

- **The clause generator**, which is responsible for the surface realization of each atomic element of the discourse specification.
- **The discourse model**, in which the discursive context for each realization is maintained. A detailed presentation of the information maintained in this discourse model is given in section 7.3. where we review the discursive constraints in EPICURE. At this point, all we need to know about this model is that it includes a hierarchy of focus spaces[62] whose structure reflects the hierarchical structure of the text plan.

The task of the discourse generator is to dispatch elements of the discourse specification to the clause generator while maintaining the discourse model. It does so by traversing the discourse specification breadth-first. During this traversal, the discourse generator does two things: (1) it stacks a focus space onto the discourse model each time it goes down one level in the hierarchical discourse specification, and (2) it calls the clause generator on each leaf it encounters. These leaves are instructions to the discourse generator to either describe a Generalized Physical Object in the ingredient list or to request an atomic Eventuality.

The discourse generator realizes each Generalized Physical Object description by a referring nominal (e.g. *two sticks of celery*, *some fresh ground black pepper*) and each atomic Eventuality request by a simple imperative clause (e.g. *Add the vegetables*, *Saute them*). Hence, EPICURE maps content unit types and Surface Linguistic Unit types in a one-to-one fashion. Eventualities have Generalized Physical Object as participant roles. Each Eventuality request has a *participants* feature whose value is a list of embedded feature structures representing Generalized Physical Objects. Each time the clause generator encounters such an embedded feature structure, it calls itself recursively. Each call results in the generation of a referring nominal as one constituent of the requesting clause under generation. The clause generator interacts with three other components of the system.

- *The discourse model*: The interaction between the clause generator and the discourse model is bidirectional. During the generation of a clause, the clause generator uses the structure of the preceding discourse recorded in the discourse model as a source of constraints on surface realization choices. After the generation of a clause is completed, the clause generator updates the discourse model.
- *The hearer model*: Aside from the list of cooking actions known to the hearer used by the discourse planner, the hearer model also contains a set of axioms modelling the hearer's capability to infer the existence of domain entities not explicitly mentioned in the previous discourse. These axioms are used by the clause generator to constrain the realization of nominals. In particular, as we will see in section 7.3.5 it allows EPICURE to use *definite* descriptions for *initial* reference to *inferrable* domain entities.
- *The domain modeller*: Before its starts to realize an Eventuality request it receives in input from the discourse generator, the clause generator passes it to the domain modeller. This allows the domain modeller to reflect the effects of the requested action onto the world model. With this mechanism, the discourse generator has knowledge of *both* the state of the world *before* the action to be requested took place and the state of the world *after* the action *after* the action to be requested took place. This allows EPICURE to generate clauses containing references to entities that does not exist until the action requested by the clause is carried out (e.g. The entity referred to by *the stalks* in the clause *Remove the stalks from the strawberries* does not exist before the action requested by the clause has taken place.)

The partition of the different generation subtasks among the components of EPICURE is the following:

[62] see section 7.2 for a definition of focus spaces.

- *Content determination* is carried out essentially by the discourse planner. Although all text planning occurs before any clause is generated, the clause generator can however be considered to perform some content determination, since it is responsible for determining what information to convey *explicitly* in the referring nominals it generates. It does so using constraints from both the discourse model and the hearer model.
- *Discourse organization* is carried out by the discourse planner. The discourse generator is *not* involved in discourse organization since it does not perform any plan reorganization but simply passes to the clause generator the realization instructions in order, as it traverses the discourse specification breadth-first.
- *Surface realization* is carried out by the clause generator, with help from the discourse generator. By maintaining the discourse model, the discourse generator allows the clause generator to use constraints from the *global* discourse structure to *locally* perform surface realization, one clause (or nominal) at a time.

5.6.2 Surface realization in EPICURE

In EPICURE, lexical choice is performed by the clause generator under a combination of discursive and encyclopedic constraints. The clause generator has access to the discursive constraints by querying the discourse model in which they are maintained. The encyclopedic constraints are given to the clause generator in the feature structure specifying a domain entity that it receives in input from the discourse generator. Internally, the generation of a clause from this **Entity Specification** is carried out by a pipeline of four processes:

1. Rewriting of the Entity Specification into a **Deep Semantic Structure**.
2. Rewriting of the Deep Semantic Structure into a **Surface Semantic Structure**.
3. Enrichment of the Surface Semantic Structure through unification with the grammar.
4. Linearization of the enriched Surface Semantic Structure into an Surface Linguistic Unit.

The third of these processes carries out morpho-syntactic grammaticalization using classic functional unification [Kay 79] implemented in PROLOG on-top of the structural unification built in PROLOG. The first two carry out lexicalization and semantic grammaticalization. We thus focus on these in this survey. Both the Deep Semantic Structures and the Surface Semantic Structures are feature structures. The features of the Surface Semantic Structure are the lexico-grammatical features a unification grammar expects to receive as input. Both the features of the Entity Specification and those of the Deep Semantic Structure are semantic. However, those of the Entity Specification are planning-oriented and more domain-specific, whereas those of the Deep Semantic Surface are realization-oriented and more domain-independent. Both the Entity Specification -> Deep Semantic Structure and the Deep Semantic Structure -> Surface Semantic Structure mappings are many-to-many. Figure 5-17 gives an example of Entity Specification from which two Deep Semantic Structures, Surface Semantic Structures and ultimately Surface Linguistic Units are generated.

During the realization of an Eventuality by a clause, the only clause constituents for which EPICURE performs an actual choice are the nominals referring to the participant roles of the Eventuality. The others are in one-to-one correspondence with the parts of the Eventuality specification they realize. In particular, EPICURE does not perform any verb choice. However, EPICURE is able to microcode a wider variety of referring nominals than any other existing system. In its full generality, the task of microcoding a nominal referring to an object token (represented by a list of attribute-value pairs) entails the four following subtasks:

```
Example of Entity Specification,
ES: ((index i1)
         (state s1)
         (specification ((substance avocado)
                         (structure individual)
                         (packaging ((shape avocado)
                                     (size regular)))
                         (ripe +))))

Deep Semantic Structure of the indefinite reference to ES
A ripe avocado,
DSS1: ((index o11)
           (semantic ((discursive-status ((given -)))
                      (specification ((agreement ((countable +)
                                                  (number singular)
                                                  (gender neuter)))
                                      (type ((category avocado)
                                             (properties ((size regular)
                                             (ripe +))))))))))

Deep Semantic Structure of the pronominalized reference to ES
it,
DSS2: ((index o13)
           (semantic ((discursive-status ((given +)
                                          (centre +)
                                          (obligatory +)))
                      (specification ((agreement ((countable +)
                                                  (number singular)
                                                  (gender neuter)))
                                      (type empty))))))

Surface Semantic Structure corresponding to DSS1,
SSS1: ((index o111)
           (semantic ((discursive-status ((given -)))
                      (specification ((agreement ((countable +)
                                                  (number singular)
                                                  (gender neuter)))
                      ((description ((head avocado)
                                     (modifier ((head ripe))))))))))

Surface Semantic Structure corresponding to DSS2,
SSS2: ((index o13)
           (semantic ((discursive-status ((given +)
                                          (centre +)
                                          (obligatory +)))
                      (specification ((agreement ((countable +)
                                                  (number singular)
                                                  (gender neuter)))
                                      (description empty))))))
```

Figure 5-17: Internal levels of representation in EPICURE's clause generator

1. Identifying the type of reference the nominal under construction is to perform.
2. Choosing what attributes of the referred token to explicitly mention in the nominal.
3. Mapping each chosen attribute onto a nominal constituent.
4. Mapping the value of each attribute onto a lexical item.

EPICURE tackles the issues raised by the first three of these tasks, but avoids those raised by the fourth as it maps attribute values onto lexical items in a one-to-one fashion. It carries out the first of these tasks during the Eventuality-Specification -> Deep-Semantic-Structure mapping, and the second and third during the Deep-Semantic-Structure -> Surface-Semantic-Structure mapping. Choosing what attributes to explicitly mention in a reference directly follows from identifying the desired type of the reference and this identification is necessarily grounded on some taxonomy of references. There are many possible reference taxonomies, but in the context of text generation they all necessarily take into account *discursive* constraints along with encyclopedic constraints. In EPICURE's reference taxonomy the high-level distinctions are grounded on discursive constraints. This taxonomy is thus presented in chapter 7 devoted to these constraints. It is also in chapter 7 that we discuss the microcoding of *different discursive* types of references. Encyclopedic constraints introduce lower-level distinctions inside some discursive types of reference. In the reminder of this section, we present these distinctions as we discuss the microcoding of *different encyclopedic* types of a *given discursive* type of reference: references to **brand-new** entities. Roughly speaking, an entity is brand-new to the discourse when it is introduced both for the first time and independently of any other entity[63]. EPICURE systematically uses an indefinite NP to refer a brand-new entity. Moreover, this indefinite NP systematically includes explicit mention of *all* the attributes of the input token. This is adequate for EPICURE because:

- The single purpose of its encyclopedic knowledge base being to support text generation, the only information available about a Generalized Physical Object is the information needed to produces unambiguous referring expressions to that object.
- In the cooking domain, object tokens typically have a very small number of properties.

In domains where objects tokens may have a large number of properties or in a natural language interface to an expert system where the generator use an encyclopedic knowledge base built essentially for reasoning purposes, the problem of choosing what attributes of an object token to explicitly mention in a brand-new reference might be non-trivial.

The various types of indefinite NPs that EPICURE uses for brand-new references to Generalized Physical Objects are directly dependent of the ontological type of these objects. In Dale's ontology of cooking entities, Generalized Physical Objects are first subcategorized into **Individuals**, **Masses** and **Sets**. Sets are in further subcategorized into **homogeneous sets** and **heterogeneous sets**.

In EPICURE's knowledge base, individuals are characterized by the *substance* they are made of, by their *size* and by their *shape*. If both shape and size are standard, the individual is referred to by a simple NP whose head is the substance (e.g. *an onion*). If the shape is non-standard it is mapped onto the NP head with the substance mapped onto either a premodifying adjective (e.g. *an onion ring*) or a postmodifying PP (e.g. *a ring of onion*). A non-standard size is expressed by premodifying adjective (e.g. *a large onion*). In addition to their substance, shape and size, individuals can also have an arbitrary number of binary valued properties. These properties are mapped onto premodifying adjectives which are ordered following an algorithm implementing the linguistic constraints described in [Ney 83] (e.g. *one hot red pepper*).

Masses are distinguished from Individuals in that they are shapeless (and thus also sizeless). Therefore they are entirely characterized by their substance and by the *quantity* of that substance. In EPICURE, that quantity can be specified either exactly or by range. In both cases, it is mapped onto the head of the NP with the substance in postmodifying PP position (e.g. *two ounces of rice* and *2-3 cups of milk*). Quantity can also be unspecified in which case the substance appears in head position premodified by the indefinite

[63] see the presentation of EPICURE's reference taxonomy of chapter 7 for an exact definition of brand-new entities.

determiner *some* (e.g. *some milk*). Finally, because substances are hierarchically organized into EPICURE's encyclopedic knowledge base, some of them are mapped (in a one-to-one fashion) onto multiple-word lexical items (e.g. in the reference *some 2% lowfat milk*, the head is the *2%-LOWFAT-MILK* substance located below *LOWFAT-MILK* itself below *MILK* in the substance hierarchy).

Homogeneous sets (i.e. sets of Individuals made of the same substance), are characterized in intention by means of a *generic element* and either a *cardinality* or an overall *quantity* of substance. The references to homogeneous sets are constructed in a similar way as those for masses, with the embedded description of the generic element in place of the substance (e.g. *Three hot red peppers*, *Two pounds of hot red peppers*). Heterogeneous sets (i.e. sets of Generalized Physical Objects made of distinct substances) are characterized in extension by the list of their elements. References to each element are grouped in a conjunctive nominal which either stands alone (e.g. *salt and pepper*) or appears in postmodifying PP position whenever an overall quantity is specified for the heterogeneous set (e.g. *350 grams of raisins and sultanas*).

Let us now evaluate EPICURE's scheme for wording brand-new references to object tokens. EPICURE is the only system to the date to generate the following types of references:

- References to masses.
- References to sets specified in intension.
- References which mention global properties of sets specified in extension.

Also, among text generators microcoding nominals, EPICURE is the only one in which the mapping between object token attributes and NP constituents is:

- Many-to-many.
- Dependent on the attributes' values.
- Based on ontological principles grasping, albeit simplistically, some pervasive naive physics distinctions.

These strengths of EPICURE's scheme to word brand-new references largely stems from its input representation formalism. As we will see in section 7.3.5, this formalism also contributes to the quality of EPICURE's scheme to the wording *non* brand-new references. However, although quite adequate in the domain of cooking, this formalism suffers from several limitations for more general encyclopedic knowledge representation purposes. In particular it does not allow the representation of:

- Contrastively or negatively defined properties. This precludes the generation of referring expressions such as *The other ones* or *One cup of non sugar-coated dried raisins*.
- Disjunctive sets. This precludes the generation of referring expressions such as *one teaspoon of salt or one tablespoon of soy sauce*.
- The various whole-part and part-part relationships of composite objects that can be assembled and disassembled. In domains involving pieces of equipment which are so popular in the generation literature, these relations are part of the core of the input representation issues.
- The functional role of an object either as an action participant or as part of a composite object. This functional role is also a crucial aspect of an object description in some domains.
- More than a closed set of non binary object properties. This drastically limits the potential variety of Surface Linguistic Units that can be used to lexicalize object properties, because only a very limited set of Surface Linguistic Unit types, (practically only adjectives, nouns or participles premodifying the head of the referring NP) can lexicalize binary properties. In particular, it precludes attributive and equative clauses commonly used for such purpose. For example if the property that a negotiation is *tense* is represented in the generator input by a pair

[*tense* +] it will be expressible only as part of an NP referring to the negotiation, as in: *The <u>tense</u> negotiation between the South African government and the ANC resumed yesterday at the Cape Town*. It will not be expressible by any of the following Surface Linguistic Units:

- Separate equative clause, as in *The negotiation between the South African government and the ANC resumed yesterday in Cape Town. <u>The atmosphere of the negotiation was tense.</u>*
- Subordinating attributive clause, as in *The negotiation between the South African government and the ANC, which resumed yesterday in Cape Town, <u>had a tense atmosphere.</u>*
- Subordinated PP, as in *The negotiation between the South African government and the ANC resumed yesterday at the Cape Town <u>in a tense atmosphere.</u>*

Such Surface Linguistic Units can be generated only from a representation where *tense* is not a binary valued property but one value of the multi-valued property *atmosphere*.

Let us emphasize that although some of the issues related to the above list of limitations have been investigated either from a natural language understanding perspective or from a problem-solving perspective, the investigation of each of them from the generation perspective constitutes another direction for future research.

5.7 Summary

The first encyclopedic constraint on the lexicalization of a token is the piece of content this token embodies. Others include the various relations of the token to lexicalize with the other tokens of the underlying domain, the knowledge of the hearer about that token etc. In almost all generator Encyclopedic Knowledge Base, a top-level distinction is made between, on the one hand, tokens representing events (or processes, actions, states, relations etc) of the underlying domain and, on the other hand, tokens representing objects (or sets, classes etc) of that domain. Typically, the first are represented by predicative structures with some nuclear roles and some satellite roles, whereas the second are represented either atomically or by predicative structures with satellite roles only. In existing generators, event tokens are invariably realized by clauses and object tokens are invariably realized by nominals. This simplifying assumption, although significantly limiting expressive flexibility, allows decomposing the overall lexical choice task into two much simpler tasks: (1) choosing clauses realizing event tokens and (2) choosing NPs realizing object tokens. Each of the generators we surveyed provides some interesting solution for only one of these two tasks. BABEL, Danlos' generator, PHRED and ANA all focus on the first test whereas EPICURE focuses on the second.

The lexical choice issue BABEL addresses is the following: given a token of an event concept general enough to be a domain-independent semantic primitive, how can a generator choose from a set of semantically distant verbs, the most appropriate for this particular token. The only encyclopedic constraint BABEL uses to perform this choice is the nuclear role values of the token. For each semantic primitive, this constraint is implemented by a discrimination network, whose arcs contains test on the token role values.

Using the same implementation technique, discrimination networks, Danlos' generator performs another type of lexical choice: choice of semantically close clause patterns to lexicalize event tokens in a very restricted domain. Any part of the conceptual content description, given as input to Danlos' generator can be tested by the arcs of its discrimination networks. Thus, compared to BABEL, Danlos' generator constrains its lexical choice on a larger chunk of the ESE.

As opposed to all the other systems we surveyed, PHRED does not have a dedicated process to perform lexical choice. Lexical choice, grammatical choice and linearization are intergrated within a single uniform surface realization process. This process is implemented by a technique very similar to functional unification except that it incorporates a special hashing-scheme to reduce backtracking. The advantage of integrating all surface realization choices is that it makes the implementation of interaction between lexical choices and grammatical choices easier. PHRED performs the same type of domain restricted clause pattern choice as Danlos'generator, but like BABEL the only encyclopedic constraint it uses to perform this choice is the content of the token to lexicalize.

ANA differs from all other systems we surveyed by two characteristics: its level of macrocoding and its input. The clause patterns ANA chooses are much more instantiated (or macrocoded) than those chosen by Danlos' generator or PHRED. Focusing on how to *combine* clauses instead of how to *construct* them, allows ANA to generate a much wider variety of clause types (though at the cost of requiring more work in hand-coding the lexicon). ANA's input is not conceptual knowledge structures but numbers from an on-line statistical database. Such numerical input necessitates an additional generation subtask, data abstraction, to be carried out. This tasks consists in abstracting clause-sized content units called messages from the flow of numerical data. In one sense, part of the overall lexical choice process is actually carried out during data abstraction. The rest is carried out during surface-realization *per se*, when messages are mapped onto clauses. Because its input is *numerical*, ANA is deprived of the possibility to bury some choices in this input and must therefore deal with the full-extent of the lexical choice issues it is addressing (although it avoids addressing a number of them which are buried in its phrasal lexicon). Data abstraction is carried out in part by a numerical C program and in part by OPS5 production rules. Surface realization is implemented by OPS5 production rules pattern matching messages onto the macrocoded clause-patterns stored in ANA's lexicon. The only encyclopedic constraint taken into account during this pattern matching is the content of the message. Thus, in this regard ANA is more like BABEL or PHRED than like Danlos' generator.

BABEL, Danlos' generator, PHRED and ANA share four characteristic:

1. They focus on the choice of verbs or of clause patterns to lexicalize event tokens.
2. Their ESE model represents *only* domain knowledge.
3. This domain knowledge is *entirely* represented by two simple inheritance hierarchies: one for events and one for objects.
4. This domain knowledge is static (i.e. it remains unchanged during the generation of an utterance).

EPICURE differs from them with respect to all of these four characteristics:

1. It focuses on choosing nominals to lexicalize object concepts.
2. Although it also contains two simple inheritance hierarchies one for object *substances* and one for event *substances*, the nodes in these hierarchies are atoms (as opposed to predicate structures with roles), and there is more to EPICURE's domain knowledge than just these two hierarchies: the plan library, the domain modeller and the world model.
3. In addition to domain knowledge, EPICURE's ESE model also incorporates knowledge of the hearer's domain knowledge.
4. EPICURE's ESE model dynamically changes during the generation of the text.

The type of encyclopedic constraint EPICURE focuses on is the structure of various types of objects and sets. The type of encyclopedic constraints Danlos' generator focuses on is direct causality between events. Neither BABEL, PHRED nor ANA focuses on any particular type of encyclopedic constraints, but for opposite reasons. BABEL because its domain is too general, PHRED and ANA because their domain are too specific.

6. Interpersonal constraints

In this chapter, we review interpersonal constraints on lexical choice. As we saw in section 2.2, these constraints include many different factors such as the communicative goals of the speaker, the social relationship between the speaker and the hearer, etc. Constraining lexicalization by such factors involves two subtasks:

1. The definition of a model of the Interpersonal Situation of Enunciation (ISE),
2. The use of such an ISE model to constrain lexical choice.

Although several NLP systems incorporate some kind of ISE model, to the best of our knowledge, only a single text generator[64] uses such a model to constrain lexicalization: PAULINE [Hovy 87], presented in the next section.

6.1 PAULINE

6.1.1 An overview of PAULINE

[Hovy 87] presents PAULINE, a system that generates paragraph-length reports of subjectively loaded events (e.g. demonstrations, elections) in a given ISE. An example of a report generated by PAULINE is given in figure 6-1.

As a reminder to Yale University to divest from companies doing
buisiness in South Africa, a large number of students constructed a
shantytown - named Winnie Mandela City - on Beinecke plaza in early April.
At 5:30 am on April 14, Yale had official destroy it; also at that time,
the police arrested 76 students. A large number of local politicians and
faculty members gave the students permission to reassemble it.

Figure 6-1: An example of text generated by PAULINE

As input, PAULINE takes the three following data structures:

- A set of **Memory Organization Packets (MOPs**) representing the event token to report about in the text. MOPs are frame-like structures [Schank 82] proposed for the representation of concepts in human memory. The main difference between MOPs and a traditional frame formalism is that in MOPs, event tokens are represented as instances of Conceptual Dependency (CD) primitives[65]. This part of PAULINE's input is very similar to that of BABEL (cf. section 5.2) except that it is paragraph sized in PAULINE[66] as opposed to single

---

[64]Recall from section 2.2 and chapter 5, that we classified the encyclopedic knowledge of the hearer, or the mutual beliefs of the speaker and the hearer about the domain as encyclopedic constraints and that, as such, we reviewed their influence on lexicalization in chapter 5.

[65]The notion of CD primitive was introduced in section 5.2.

[66]In his thesis, Hovy does not give any example of a complete set of MOPs representing a paragraph-sized event token. He does however, give examples of sentence-sized event tokens. Such sentence-sized event tokens are certainly constituents of larger event tokens in PAULINE's input. One of this example can be found in figure 6-5 of the next subsection discussing PAULINE's surface realization process.

sentence sized in BABEL.

- A set of binary or ternary features modelling two aspects of the ISE in which the event token is to be reported by PAULINE: the conversation setting and the interpersonal goals of PAULINE toward the hearer. The features representing the conversation setting, along with their possible values, are given in figure 6-2. The features representing the interpersonal goals of PAULINE toward the hearer, along with their possible values, are given in figure 6-3.
- A list associating each subtoken of the top-level input event token with a ternary mark: GOOD, BAD or NEUTRAL. This list represents a third aspect of the ISE: PAULINE's bias concerning the entities in the event to report. For example, such bias marks can be associated with a male participant of the event. It can then be used to choose between the expressions *That jerk* and *That gentlemen* to refer to him.

Conversational atmosphere:

- time = *much, some or little*
- tone = *formal, informal or festive*
- conditions = *good or noisy*

Speaker:

- knowledge of the topic = *expert, student or novice*
- interest in the topic = *high or low*
- opinions of the topic = *good, neutral or bad*
- emotional state = *happy, angry or calm*

Hearer:

- knowledge of the topic = *expert, student or novice*
- interest in the topic = *high or low*
- opinions of the topic = *good, neutral or bad*
- emotional state = *happy, angry or calm*
- language ability = *high, normal or low*

Speaker-hearer relationship

- depth of acquaintance = *friends, acquaintances or strangers*
- relative social status = *dominant, equal or subordinate*
- emotion = *like, neutral or dislike*

Figure 6-2: Features modelling conversation setting in PAULINE

In addition to the ISE model represented by these features, PAULINE also incorporates a model of the Encyclopedic Situation of Enunciation (ESE). This model is encoded by two inheritance hierarchies: one for event concepts and one for object concepts. The element of these hierarchies are represented by MOPs. The top-level elements of the event conceptual hierarchy are CD primitives and the leaves of this hierarchy

Hearer:

- affect hearer's knowledge = *teach, neutral or confuse*
- affect hearer's opinions of topic = *switch, none or reinforce*
- involve hearer in the conversation = *involve, neutral or repel*
- affect hearer's emotional state = *anger, neutral or calm*
- affect hearer's goals = *activate, neutral or deactivate*

Speaker-hearer relationship:

- affect hearer's emotion toward speaker = *respect, like or dislike*
- affect relative status = *dominant, equal or subordinate*
- affect interpersonal distance = *intimate, close or distant*

Figure 6-3: Features modelling PAULINE's interpersonal goals

are event tokens similar to those appearing in the input. Since parts of these tokens may be conveyed in the report in addition to those appearing in the input, they carry a default bias mark as either GOOD, BAD or NEUTRAL. Thus, in addition to encyclopedic constraints, some interpersonal constraints are also encoded in these two conceptual hierarchies.

PAULINE's architecture is as a pipeline of four processes:

- *Pragmatic set-up*: Trying to use the features of PAULINE's ISE model *directly* as a source of constraints on generation choices, would amount to answering questions such as "how does the fact that the hearer is socially dominant over the hearer constrains sentence length ?" or "how does the fact that the speaker has the goal of being friendly toward the hearer constrain voice choice ?". The fact that these questions are clearly ill-formulated, indicates that the features of PAULINE's ISE model have to be used *indirectly*. The need for such indirection justifies the process of pragmatic set-up during which, from these features, PAULINE derives a set of **rhetorical goals (RGs)**. These RGs embody stylistic factors which *can* be used to *directly* constrain generation choices. Each RG is represented by a feature with two to four possible values. PAULINE's RGs together with their possible values are given in figure 6-4. A given set of input features representing a particular conversation setting and a particular set of interpersonal goals toward the hearer, activates some RGs during pragmatic set-up. The activated RGs are then used as interpersonal constraints on a wide range of decisions in each of the subsequent processes: topic collection, topic organization and textual realization.
- *Topic collection*: During topic collection, PAULINE attempts to relate the input event token with those stored in its ESE model. It decides at a high-level what parts of that input event to convey in the report. It also decides whether to mention parts of stored event tokens similar to the input one. The output of the topic collection process is an unordered pool of content units to potentially convey in the text.
- *Topic organization*: During topic organization, PAULINE groups and order the various content units it selected during topic collection. The output of topic organization is an ordered list of sentence-sized content units.
- *Textual realization*: During textual realization, PAULINE linearly scans the list of content units resulting from topic organization, and for each of them, produces a corresponding English sentence.

The last three of the four above processes almost exactly correspond to the three generation subtasks we defined in section 1.1. In particular, PAULINE carries out all discourse organization during the topic collection process, and all surface realization during the textual realization process. However, although PAULINE carries out most content determination during the topic collection process, some low-level content determination also occurs during the texual realization process. During this process, the activated RGs might indicate that some content unit slots should be omitted. We give an example of this process of low-level content determination under interpersonal and surface realization constraints in the next section where we describe PAULINE's textual realization in further details.

simplicity = simple, normal or complex
verbosity = terse, normal or verbose
formality = highfalutin, normal or colloquial
force = forceful, neutral or quiet
floridity = dry, neutral or flowering
timidity = timid or reckless
partiality = impartial, implicit or explicit
detail = details, interpretations or both
color = facts only or with color
openmindness = yes or no
flightiness = flighty or stuck
haste = pressured, unplanned, somewhat planned or planned
respect = arrogant, neutral, respectful or cajoling
aggression = aggressive, neutral or placating
incitement = inciting, normal or calming
number of personal reference to the hearer
number of personal reference to the speaker (i.e. to PAULINE)

Figure 6-4: PAULINE's rhetorical goals

6.1.2 Surface realization in PAULINE

In PAULINE all surface realization subtasks, lexicalization, semantic grammaticalization, morpho-syntactic grammaticalization and linearization are carried out simultaneously by a single process. During this process, some low-level content determination also occurs. This process is implemented by a set of LISP functions called **Syntax Specialists**. These functions intertwine control information with the linguistic data often represented declaratively in the grammar and the lexicon of other systems. In PAULINE, no distinction is made between general constituent ordering patterns usually stored in the grammar (e.g. the <WH-SUBJECT VERB OBJECT> pattern for simple question clauses) and partially lexicalized patterns (e.g. < X kick the bucket >) usually stored in phrasal lexicons, nor between such phrasal patterns and individual words. They are all viewed as partially realized lexico-grammatical patterns of various size and specificity. The typical task of a Syntax Specialist is to further realize such a partially realized pattern under a combination of grammatical, interlexical, encyclopedic, discursive and interpersonal constraints. In PAULINE, all these contraints are integrated in the code of the Syntax Specialists. There are three main kinds of Syntax Specialists:

- Linguistic Syntax Specialists, such as SAY-NOUN-GROUP, which are indexed exclusively by the type of SLU they can use to realize a content unit, regardless of the encyclopedic type of the content unit.
- Conceptual Syntax Specialists, such as SAY-AGE, which are indexed exclusively by the

encyclopedic type of content unit they can realize regardless of the SLU types they use to do it.

- Hybrid Syntax Specialists, such as SAY-EVENT-SENTENCE, which are indexed by *both* the encyclopedic type of the content unit they can realize and the SLU they use to do it.

PAULINE realizes a sentence-sized content unit by linearly scanning a realization stream of **Syntax Goals** from left to right. These Syntax Goals are described by three features:

- A Syntax Specialist.
- A content unit to realize using that Syntax Specialist
- The grammatical and/or conceptual context of the realization.

For each Syntax Goal it encounters, PAULINE calls the Syntax Specialist of the Syntax Goal with the content unit and context of the goal as parameters. The result of this call is a list containing lexical items and/or lower-level Syntax Goals. This list replaces the input Syntax Goal in the realization stream. The content units of lower-level Syntax Goals are those embedded units of the content unit of the higher-level Syntax Goal that need further realization. Thus, PAULINE recursively realizes lower and lower level content units by lower and lower level Surface Linguistic Units in a depth-first fashion. At the beginning of this process, the realization stream contains a single top-level syntax goal whose Syntax Specialist is SAY-SENTENCE, whose content unit is a sentence-sized event token and whose context is *NIL*. At the end of the process the realization stream is a linearized and morphologicalized string of lexical items, i.e. an English sentence. An example of sentence realization is given in figure 6-5. This example illustrates a case where PAULINE decided to omit some slots of the input token, namely the manner of MTRANS-6, and the name and sex of SUE, during surface realization. This decision is based on some activated RGs[67].

The various choices performed during the surface realization process are constrained by PAULINE's ISE model. When performing a choice, the Syntax Specialists take into account both the bias marks inside the content unit under realization and the activated Rhetorical Goals derived from the input representation of the conversational setting and of PAULINE's interpersonal goals toward the hearer. A wide variety of surface realization choices are influenced by these interpersonal constraints. This variety is facilitated by the fact that PAULINE does *not* distinguish between different types of decisions, but implements them uniformly by call to Syntax Specialists. Excerpts from texts illustrating realization in PAULINE varies with the parameters of its ISE model are given in Figure 6-6. Both texts derive from the same content but from a different set of values for the parameters in the ISE model. In particular, during the pragmatic set-up phase of generation, these parameters activate the Rhetorical Goal FORMALITY with a HIGHFALUTIN value in the case of text (1) and with a COLLOQUIAL value in the case of text (2). During the textual realization phase, these different values for the Rhetorical Goal FORMALITY results in the selection of different options for various types of lexical choice. As highlighted in figure 6-6 the HIGHFALUTIN FORMALITY favors:

- Lenghty proper names over short ones (e.g. *The Empire of Japan* vs. *Japan*).
- NPs and PPs over clauses (e.g. *in discussion* vs. *talking*, *the maintenance of peace* vs. *having peace*).
- Specific nouns over more general ones (e.g. *solicitation* vs. *request*).
- A minimal use of pronominalization (e.g. *that nation* vs. *them*) and ellipsis (e.g. *Its Government and its Emperor* vs. *Its Government and Emperor*).

[67]Exactly which one is not specified by Hovy.

```
Example of sentence-sized content unit:
((token MTRANS-6)
 (concept-type action)
 (concept mtrans)
 (actor JIM)
 (object ((token DEATH-10)
          (concept-type state-change)
          (concept health-change)
          (actor Janet)
          (from alive)
          (to dead)))
 (from JIM)
 (to ((token SUE)
      (concept-type object)
      (concept person)
      (age ((token SUE-AGE)
            (concept-type measure)
            (unit year)
            (number 23)))
      (name Sue)
      (residence New-Haven)
      (sex female)
      (size small)))
 (manner quiet))

Initial realization stream:
  <{Syntax-Goal = [Syntax-Specialist = SAY-SENTENCE
                   Topic = MTRANS-6
                   Context = NIL]}>

Final realization stream:
  <The small 23-years old from New-Haven was told by Jim that Janet
   kicked the bucket>

One intermediate realization stream:
  <The
   small
   {Syntax-Goal = [Syntax-Specialist = SAY-AGE
                   Topic = SUE-AGE
                   Context = HEAD]}
   {Syntax-Goal = [Syntax-Specialist = SAY-POST-NOUN-MODS
                   Topic = SUE
                   Context = (residence name)]}
   {Syntax-Goal = [Syntax-Specialist = SAY-PREDICATE
                   Topic = MTRANS-6
                   Context = NIL]}>
```

Figure 6-5: Example of surface realization in PAULINE

In addition to the Rhetorical Goals activated by the parameters representing the conversational setting and the interpersonal goals toward the hearer in PAULINE's ISE model, the bias marks representing PAULINE's opinion on the participants of the event to report are also used to constrain lexical choice. For example different values for a bias mark associated with a human male participant result in different referring NPs: A GOOD value may result in the premodification of the head of the referring NP by *That jerk*, a BAD value may result in the premodification of the head of the referring NP by *That gentleman* and wheras a NEUTRAL value results in no such premodification.

(1): *Yesterday, December 7, 1941, the United States of America was suddenly and deliberatly attacked by naval and air forces of* **the Empire of Japan**. *The United States was at peace with* **that nation** *and, at the* **solicitation** *of Japan, was still* **in conversation** *with its Government and* **its** *Emperor looking forward to* **the maintenance of peace** *in the Pacific.*

(2): *We were suddenly and deliberatly attacked by naval and air forces of* **Japan** *yesterday, December 7, 1941. We were at peace with* **them**. *At Japan's* **request**, *we were still* **talking** *to their Government and Emperor. We were looking forward to* **having peace** *in the Pacific.*

Figure 6-6: Different realizations of the same input content in different interpersonal situations in PAULINE

From a lexical choice perspective, the main interest of Hovy's work is to show that a wide range of interpersonal factors *can* (and ideally *should*) be used to constrain an equally wide range of lexicalization options. The outputs of PAULINE are quite impressive in diversity as well as in quality. However, one problem with PAULINE is that the features it uses to model the ISE is totally *ad-hoc*. Not only they are not grounded in any theoretical work on natural language interaction, but Hovy does not even attempt to justify them. Thus, as they stands they are little more than an agenda of interesting directions for future research. As pointed out by [Elhadad 87] the current state-of-the-art in sociolinguistics, philosophy of language and artificial intelligence is still far from being able to propose a unified theoretical background for modelling as many aspects of the ISE as Hovy attempted to capture in PAULINE. Thus, for having a chance to be successful, future ISE models for lexicalization will have to be very restricted in terms of both the number of aspects of the ISE they attempt to capture and the scope of the underlying application domain.

Another problem with PAULINE is its implementation of surface realization. Intertwining grammatical and linguistic data with control information inside LISP code, makes extensions or modifications extremely difficult. Integrating all the different types of constraints on surface realization into a single unstructured procedural representation practically precludes the concurrent acquisition of these constraints by several developers in a systematic way. It also encourages the development of ad-hoc constraints vs. the abstraction of more general, principled ones. In short, implementing a surface generator purely proceduraly by a set of LISP functions in which implicit control information is scattered hardly scales-up. So, although PAULINE's outputs are impressively varied, this variety stems more from a great amount of hand-work during knowledge acquisition, than from the flexibility of PAULINE's architecture or implementation methodology.

7. Discursive constraints

7.1 Introduction

In this section we survey the influence of the Discursive Situation of Enunciation (DSE) on lexical choice. Modelling the influence of the DSE on the generation of an utterance raises two distinct issues: (1) the representation of the DSE, and (2) its use as a constraint on generation decisions. Depending on the application at hand, the DSE of an NLG system is either *conversational* or *textual*. The generation of one or two sentence long conversation turns as part of a Question-Answering System as an example of conversational DSE. In contrast, the generation of a one or two paragraph long connected text, is an example of textual DSE. Although several QASs maintain a model of the ongoing conversation, they primarily uses it to constrain content of a response not lexical choice. These systems are thus not of direct interest to us. A good survey can be found in [Paris 85].

The representation of the DSE consists of discourse the various units that make up a discourse and the relations binding them together. Discourse units can be of various size, from the paragraph to the individual word, and a great variety of relations can bind them together as a discourse. These relations can link the units content or their surface form, they can be functional or structural etc. Each of these relations provides one dimension along which to describe the underlying discourse structure. There is a considerable body of work dedicated to this topic in both computational and non-computational linguistics, and many models proposed a multi-dimensional description of discourse structure (see for example the three dimensions proposed by [Grosz & Sidner 86] and the five dimensions proposed by [Elhadad 90]). An excellent survey of the literature on textual DSE modelling can be found in Chapter 2 of Dale's thesis [Dale 88]. In this survey we present exclusively work where implemented DSE models constrains lexical choice in text generators. In the next sections of this chapter we first present the notion of **focus of attention**, and its use in BABEL [Goldman 75] and TEXT [McKeown 85b]. We then present the notion of **centering** and the **Given-New distinction** and the combined use of both in EPICURE [Dale 88]. Finally, we present the notion of textual **cohesion** and its use in PAUL [Granville 83].

Some lexical choices issues such as pronominalization and definite NP anaphora *must* be constrained by discursive factors. Moreover they systematically arise with any attempt to generate connected text. Thus, text generators which do *not* make DSE modelling a core issue of their design such as Danlos' system or ANA [Kukich 83], have taken some simple practical approach to make these decision *without* maintaining a DSE model based on some discourse structure theory. In these practical approaches, both encyclopedic and primitive discursive constraints are highly integrated. We thus presented them in sections 5.3 and 5.5 on encyclopedic constraints.

7.2 Focus of attention

Any discourse[68] is *about* some set of entities from some real or fictitious world. The notion of focus of attention, which originates in psycholinguistic research, is based on the intuition that, at any given point in a discourse, the speaker and the hearer *focus their attention* on some particular subset of all discourse

---

[68]Whether a conversation or a written text.

entities[69]. Moreover, as discourse unfolds, the degree to which each discourse entity is in focus dynamically changes, a phenomenon known as **focus shift**.

The first computational DSE models using this notion of focus were put forth by researchers working in Natural Language Understanding. Investigating the issue of definite anaphora resolution in task-oriented dialogs between an expert and an apprentice, [Grosz 78] proposed distinguishing between **global focus** on one hand and **local or immediate focus** on the other. As pointed out by [Dale 88], this distinction corresponds to the distinction between the hearer's memory for concepts (global focus) and the hearer's memory for surface linguistic forms (local focus). Grosz proposes a decomposition of global focus into a hierarchy of **focus spaces** patterned after the hierarchical decomposition of the task underlying the discourses she studied. She also claims that entities in global focus are preferably realized by definite NPs whereas entities in local focus are preferably realized by pronouns. Working on pronominal anaphora resolution, [Sidner 79] refined Grosz's notion of local focus. In her work, the following focus information is associated with each sentence:

- The **current focus**: the entity in focus in the sentence currently analyzed,
- The **potential focus list**: the list of entities that are candidate for the focus of the next sentence, defined as the entities mentioned in the current sentence,
- The **past foci stack**: the entities that were in focus in preceding sentences.

Sidner designed an algorithm intertwining two tasks during discourse interpretation : (1) maintenance of these three pieces of focus information by tracking focus shifts, and (2) use of this focus information to choose among various discourse entities candidate for the referent of a pronoun[70]. Focus information similar to Sidner's local focus has been used to constraint lexical choices in three NLG systems: BABEL [Goldman 75], TEXT [McKeown 85b] and PAUL [Granville 83]. We elaborate on the first two in the next subsection. Because PAUL also uses the notion of textual cohesion presented in the section 7.4, the discussion of its use of focus is delayed until then.

7.2.1 Focus constraints on lexical choice in BABEL

BABEL [Goldman 75], that we overviewed in chapter 5.1, generates single sentences from Conceptual Dependency (CD) networks. A given input network represents a particular token of a CD primitive. Each CD primitive has a set of roles associated with it. Different tokens of the same primitive differ with respect to two factors: (1) their particular role values, and (2) the specification of which role is *in focus*. Both these factors are used to constrain main verb choice for clauses realizing the input tokens. How the first of these two factors is used was explained in section 5.1. The idea underlying the use of the second is that it is a lexical property of some verbs to focus on a given element of their case role structure. Thus, knowledge of which role should be in focus can be used to choose between otherwise synonymous verbs. Consider, for example, tokens of the ATRANS (Abstract TRANSfer) CD primitive with four roles: a human H1, another human H2, an amount of money M and an indefinite amount of some goods G. Depending on which role is in focus in such an ATRANS token, the main verb of the clause realizing this token may be either:

- *to pay* if the focus is on M, yielding the clause pattern: *H1 pay H2 <u>M</u> dollars for some G*,

---

[69] i.e. all the entities the discourse is about.

[70] Once resolved, presence of a pronominal anaphora is in turn used as clue for of focus shift detection. Both tasks are thus feeding upon each other.

- *to buy* if the focus is on H1, yielding the clause pattern: *H1 buy some G from H2 for M dollars*,
- *to sell* if the focus is on H2, yielding the clause pattern: *H2 sell some G to H1 for M dollars*,

To the best of our knowledge, no other attempt has been made since BABEL to constrain verb choice using focus information. There are two things to note about BABEL's use of focus. First, using focus, Goldman models the one-to-many mapping between the semantic role structure of a token and the syntactic roles structure of the clauses realizing it (cf. section 5.1). Because this mapping is essentially dependent on the main verb of the clause, it is possible to use it as a criteria for main verb choice. However, focus is only one of several constraints on the desired syntactic role structure. The overall grammatical and lexical context surrounding the clause can also constrain this structure. The second thing to note about focus information in BABEL is that it is assumed to be given in the input. As a single sentence generator, BABEL ignores the issue of deciding when to shift or maintain focus from one sentence to the next. Let us now shift our focus to TEXT [McKeown 85b], a text generator addresses precisely this issue.

7.2.2 TEXT

[McKeown 85b] presents TEXT, a system generating paragraph-length responses to questions about the structure of a database. TEXT's architecture is a pipeline of two modules:

- The **strategic component**, responsible for both content determination and discourse organization,
- The **tactical component**, responsible for surface realization.

The tactical component is a functional unifier that produces an English text by unifying an input functional description with its functional unification grammar. The emphasis of McKeown's research was on the strategical component. This component searches a KL-ONE style Encyclopedic Knowledge Base explicitly representing knowledge about the database structure. The generation of a response starts by the selection of a pool of relevant propositions to convey in the response. Propositions are then retrieved from this pool and at the same time organized by traversing an ATN encoding a **textual schema**, i.e. a prototypical textual organization pattern identified from human-generated responses. Each arc of the ATN might select several propositions from the pool. Being able to choose one proposition per arc such that coherence[71] of the generated text is maintained (a discourse organization choice) is the primary motivation for the use of focus information in TEXT. However, this information is also passed to the tactical component of TEXT, which uses it to constrain both syntactic voice[72] (a grammatical choice) and pronominalization (a lexical choice). Therefore two aspects of TEXT are of interest to us: (1) how focus information is represented and (2) how it is used to perform pronominalization.

Using the three data structures proposed by Sidner to represent focus information (current focus, potential focus list and past foci stack), TEXT's strategical component incorporates an algorithm to decide when and how to shift focus from one proposition to the next. In addition to focus information, the algorithm also takes as input, the propositions that can be generated next. If we call the list of entities these propositions refer to ***L***, then TEXT focus shifting algorithm is as presented in figure 7-1.

When asked to realize a proposition, TEXT's tactical component has the focus information managed by

---

[71]See section 7.4 for a definition of textual coherence.

[72]i.e. passive sentence vs. active sentence.

```
IF there is an entity that is a common member of L and of the
   potential focus list
THEN make this entity the current focus, and push the current focus
     on the past foci stack (i.e. shift focus to a new entity)
ELSE IF the current focus is a member of L
     THEN do nothing (i.e. maintain focus)
     ELSE pop the past foci stack to find an element of L in it.
          IF there is one
          THEN make this element the current focus and push the
               current focus on the past foci stack
               (i.e. shift back to a past focus)
          ELSE find among the candidate propositions, the one with the
               greatest number of implicit links with the entities in the
               potential focus list,
               make the current focus the default focus of that proposition
               and push the current focus on the past foci stack.
```

Figure 7-1: Focusing algorithm in TEXT

the strategical component at its disposal. This allows TEXT to pronominalize reference to entities maintained as the current focus over consecutive sentences. Consider the sample TEXT output of figure 7-2, *its surface-going capability* is used in the second sentence, instead of, say, *The surface-going capability of the ship*, because the entity SHIP is the maintained focus over the two first sentences.

A ship *is a water-going vehicle that travels on the surface.* ***Its***
surface-going capabilities are provided by the DB attributes DISPLACEMENT
and DRAFT. Other DB attributes of the ship include MAXIMUM_SPEED,
PROPULSION, FUEL (FUEL_CAPACITY and FUEL_TYPE), DIMENSIONS,
SPEED_DEPENDENT_RANGE and OFFICIAL_NAME. The DOWNES, for example has
MAXIMUM_SPEED of 29, PROPULSION of STMTURGRD, FUEL of 810 (FUEL_CAPACITY)
and BNKR (FUEL_TYPE), DIMENSION of 25 (DRAFT), 46 (BEAM), and 438 (LENGTH)
and SPEED_DEPENDENT_RANGE of 4200 (ECONOMIC_RANGE) and 2200
(ENDURANCE_RANGE).

Figure 7-2: An example of response generated by TEXT

This simple case of pronominalization is the only kind of lexical choice constrained by focus information in TEXT. This is essentially due to the fact that TEXT was everything but an attempt to address lexical choice issues. However, the focus information maintained by TEXT's strategical component is quite rich, and a tactical component designed to perform a wide range of lexical choices could probably advantageously use this information. It is up to future research to further investigate this direction.

7.3 Discursive constraints in EPICURE

In section 5.6 we overviewed EPICURE [Dale 88], a system that generates cooking recipes. We saw the issue it emphasizes is that of microcoding nominals that realize various types of references to object tokens in the cooking domain. This microcoding is constrained by a combination of encyclopedic and discursive constraints. The encyclopedic constraints are essentially the position of the object token in Dale's ontology of Generalized Physical Objects (see section 5.6.2.). The discursive constraints are based on the three following notions:

- The notion of hierarchy of global focus spaces.
- The Given-New distinction.
- The notion of centering.

The first of these three notions was presented in section 7.2. The presentation of the two others is the object of the next two subsections. In the subsequent subsections we first present the integration of these three notions in EPICURE's discourse model. We then present the integration of (1) the information maintained in this model with (2) the encyclopedic information contained in input tokens, in the definition of EPICURE's taxonomy of references. Finally we present the microcoding of references for all elements of this taxonomy except for brand-new references. The microcoding of brand-new references, was presented in section 5.6.2.

7.3.1 The Given-New distinction

As pointed out in section 2.1, the interpretation of an utterance is very much dependent on its situation of enunciation. Among all the information the hearer uses to interpret an utterance, several linguists ([Halliday 68], [Chafe 76] among others) pointed out the necessity to distinguish between the information, **Given** by the situation and the **New** information conveyed by the utterance in itself. This distinction[73] is especially relevant for the study of reference. In order to determine what properties of the referent to explicitly convey in a referring expression, the speaker needs to make assumptions concerning the properties the hearer can somehow recover from the situation of enunciation. The more recoverable from the situation (i.e the more *Given*) information about the referent is, the less explicitly it needs to be conveyed in the reference. From the empty string realizing elliptical reference to complex NPs with premodifying open-class words and postmodifying relative clauses, there is a wide range of explicitness degrees for different types of Surface Linguistic Unit. Consequently, to choose among different types of Surface Linguisitic Unit, the assumed degree of familiarity of hearer with the referent is an essential factor. Moreover any set of distinctions modelling this factor, should meet the following criteria:

- It must be much finer than the binary Given-New distinction, namely fine enough to define a range of familiarity degrees matching the range of explicitness provided by natural language.
- It should incorporate all major aspects of the situation of enunciation.

Figure 7-3 presents a taxonomy of Given-New information due to [Prince 81] which meets these two

[73]Each linguist discussing the Given-New distinction seems to have its own definition for it. Moreover, alternative terminologies have been used by yet other authors to capture more or less similar distinctions, such as **Theme/Rheme** or **Topic/Comment**. The notion of local focus that we saw in section 7.2 and that of Centering which is the object of the next subsection aim at modelling a closely related phenomenon. It is beyond the scope of the present survey to discuss the differences and similarities or these various approaches. See [Chafe 76] and [Prince 81] on this issue. The presentation of EPICURE's discourse model integrating global focus, given-new distinctions and centering in section 7.3.3 should make these ideas clear enough for our purpose.

criteria. Although it needs to be further refined to support wording of references in an NLG system, it serves as a good starting point.

Insert here figure 2.2 p47 of Dale's thesis

Figure 7-3: Prince's taxonomy of Given-New information (from [Dale 88])

Let us now briefly define each element in this taxonomy. At some point in a discourse, an entity to be referred to is:

- **Unused** if it has not been mentioned in previous discourse, but is assumed by the speaker to be known to the hearer on the basis of some common background. The first occurrences of expressions like *the supreme court* or *the super bowl* in a discourse addressed to a hearer who has lived in the USA, are examples of references to unused entities.
- **Brand-New** if it is assumed to be unknown to the hearer. A brand-new entity can be **anchored** if it is introduced to the discourse by the expression of its relation to an entity assumed to be known to the hearer. Unanchored brand-new entities are always referred to using indefinite NPs, as in *I bought a new pen yesterday*, whereas anchored brand-new entities may be referred to using either indefinite NPs, like *a woman I met at the Opera* or definite NPs as *the guy Jennifer went out with last night*.
- **Non-containing inferrable** if it has not been mentioned in previous discourse, but its existence is assumed to be inferrable from previously mentioned entities by the hearer using his commonsense knowledge. In the following discourse: *Felix took the qualifier examination. He complained that the VLSI question was too easy*, the entity referred to by *the VLSI question* is an example of non-containing inferrable.
- **Containing inferrable** if it is to be explicitly introduced to the discourse as related to another entity. Note that this other entity may have been mentioned in previous discourse as in *the skin of the avocado* or may not as in *the kernels of a fresh ear of corn*.
- **Situationally evoked** if its existence can be derived from the interpersonal situation of enunciation. Deictic expressions such as *You* and *I* are examples of references to situationally evoked entities.
- **Textually evoked** if there is mention in previous discourse of either the entity itself or some other entity with which it shares some significant characteristics.

Aside from accounting for all three aspects of the situation of enunciation we defined in section 2.2, the main interest of this taxonomy, from a lexical choice perspective, is that the distinctions it makes are significant with respect to the surface linguistic types of referring expressions: only textually evoked entities are candidate for ellipsis, pronominalization and one-anaphora, and only brand-new unanchored entities are systematically referred to by mean of indefinite NPs. The other elements of the taxonomy subcategorizes the uses of definite NPs for initial reference.

[Dale 88] used Prince's distinction as one of the factors influencing the process of microcoding referring nominals in EPICURE. In particular, he redefined the notions of brand-new entity, of containing inferrable and of textually evoked entities in the practical context of its implementation. We give Dale's redefinition of Prince's categories in section 7.3.4 as we present EPICURE's taxonomy of references.

7.3.2 Centering

Investigating the relationship between discourse structure and pronominalization [Grosz, Joshi & Weinstein 81] put forth the notion of **centering**. The intuition behind this notion is that for each sentence in a discourse, some discourse entities have a privileged status. More precisely, a set of **forward-looking centers** and a single **backward-looking center** (or simply **center**) can be associated with each sentence. By relating the pronominalization patterns in the sample discourses they studied with the discourse entities they manually identified as the forward and backward looking centers of these discourses' sentences, they proposed the following pronominalization rule: *A pronoun should be used to refer to a discourse entity E if either:*

- *E is the maintained center of both the current and the preceding sentences*
- or *E was mentioned in previous discourse, and E differs from the maintained center Cb of both the current and the preceding sentence, and Cb is pronominalized in the current sentence.*

This notion of centering is very similar to Sidner's notion of local focus presented in section 7.2. Grosz & al's forward-looking centers correspond to Sidner's potential focus list and Grosz & al's backward-looking center corresponds to Sidner's current focus[74]. As pointed out by [Dale 88], Grosz & al's notion of centering is both simpler and more generation-oriented than Sidner's notion of focusing. Its main drawback is that Grosz & al did not propose an algorithm to determine the centers (both backward and forward looking) of a sentence. Some attempts to do so have been carried out since then by [Kameyama 86] and [Brennan, Friedman & Pollard 87]. For the restricted purpose of guiding pronominalization in EPICURE, Dale systematically defined the center of a clause as the object of the cooking action realized by the preceding clause, *in the state it is left by the execution of that action*. It is the peculiarity of its domain that allowed Dale to use such a simple definition for the center of a sentence. Other domains would undoubtedly require more sophisticated and context dependent definitions.

7.3.3 EPICURE's DSE model

EPICURE's DSE[75] model is based both on Grosz's notion of hierarchical global focus spaces and on Grosz et al's notion of center. More precisely, as it generates a recipe, EPICURE maintains the following data structure:

---

[74]Note that, for our purpose of discussing its relevance for generation, we somewhat simplified Sidner's model in our presentation. In fact what we called *current focus* was called *discourse focus* by Sidner's who was distinguishing it from the notion of *actor focus* that we omitted for simplicity sake. So, what in fact corresponds to Grosz & al's backward-looking center is Sidner's discourse focus.

[75]Discursive Situation of Enunciation, cf. section 2.2

- **The current clause workspace**: contains both the Deep Semantic Structure of the clause under construction and the corresponding Surface Semantic Structure.
- **The previous clause content**: contains the Deep Semantic Structure of the preceding clause.
- **The current center**: contains the Entity Specification of the object of the preceding clause.
- **The focus stack**: a stack of currently open focus spaces. There is one focus space per Entity Specification passed to the clause generator. Each focus space contains pointers to the world model for this Entity Specification as well as for the embedded Entity Specifications it itself contains. Focus spaces are thus organized in a hierarchy patterned after the hierarchical text plan for the recipe.
- **The closed focus space list**: When the realization of an Entity Specification is completed, its corresponding focus space is popped from the focus stack and appended to the closed focus space list.

When generating a referring expression to a Generalized Physical Object (GPO) token, the clause generator thus has access to:

- Both the semantic content and surface realization of the embedding clause,
- The semantic content of the preceding clause[76].
- The list of entities mentioned in previous discourse.

It uses this discursive information in combination with the encyclopedic type of the GPO token to choose the type of reference to make, i.e. to single out an element in its reference taxonomy. This is done during the Entity Specification -> Deep Semantic Structure mapping stage of surface realization[77]. How it is done is the object of the next subsection. During the Deep Semantic Structure -> Surface Semantic Structure mapping stage of surface realization, the surface form of the nominal appropriate for the chosen type of reference is built. How this is done for each element of EPICURE's reference taxonomy is the object of section 7.3.5.

7.3.4 EPICURE's reference taxonomy

In the taxonomy of reference underlying the microcoding of nominals in EPICURE, four main types of references are first distinguished. Three of them correspond to elements in Prince's taxonomy of Given-New information[78]: references to brand-new entities, references to textually evoked entities and references to containing inferrables.

Recall from section 7.3 that a containing inferrable is a domain entity that is explicitly introduced to the discourse in relation to another entity (e.g. *the skin of an avocado*). In the general case, various encyclopedic relations between two domain entities could license the discursive introduction of one of them as a containing inferrable. However, in the restricted domain of EPICURE only one such relation is

---

[76]i.e. the last action requested to the hearer. It includes the result of that action, i.e. the GPO token which is by definition the current center.

[77]Recall from section 5.6.3 that EPICURE carries out the lexicalization and semantic grammaticalization of tokens in two mapping stages: the mapping of the Entity Specification representing the token into a Deep Semantic Structure, followed by the mapping of the Deep Semantic Structure into a Surface Semantic Structure to be passed to the functional unification grammar.

[78]For other elements in Prince's taxonomy, although [Dale 88] discusses some mechanisms to refer to them, these mechanisms were not implemented in EPICURE (essentially because tokens of such elements cannot be represented by EPICURE's encyclopedic knowledge representation scheme (cf. section 5.6.2.)). We thus do not discuss them here.

considered, the *ancestry* relation between a GPO and the result of its transformation into another GP0 by a cooking action (e.g. the relation between a whole avocado and the avocado skin resulting from the action of peeling the avocado). This relation is simply represented by an *ancestor* feature in the feature structure representing the GPO *resulting* from the transformation. The value of this feature is the original GPO on which the transformation was applied. Whether a GPO token is a containing inferrable in EPICURE is thus determined by the presence or absence of an ancestor feature in the Entity Specification representing it.

The difference between a brand-new GPO token and a textually evoked GPO token is determined by the presence of the GPO in the discourse model. Thus, in the computational context of EPICURE's implementation, the distinction between the different elements of Prince's taxonomy of Given-New information is redefined both in terms of the information contained in the discourse model and the GPO token to refer to. More precisely, a GPO token is:

- **Containing Inferrable** if it has an ancestor feature (regardless of its presence in the discourse model).
- **Textually evoked** if (1) it has no ancestor feature and (2) it is present in the discourse model.
- **Brand-new** if (1) it has no ancestor feature, (2) it is not present in the discourse model (i.e. if it has not been already mentioned), and (3) in the case of individuals, if its substance differs from the substance of the last GPO token generated (this substance is part of the previous clause content recorded in the discourse model).

The fourth main category of reference in EPICURE includes references to tokens of individuals (without an ancestor feature) which are not present in the discourse model but have the same substance as the last GPO token generated. This last property licenses EPICURE's use of one-anaphora in nominals referring to them. Whether tokens of this type fall into Prince's category of new (although not brand-new) entities or Prince's category of textually evoked entities is debatable. They can be considered new in the sense that they have not been mentioned themselves in previous discourse and they have not come to be known to the hearer by any extra-textual means either. However, a notion of textual evocation restricted to that of direct previous mention is impoverished. It seems intuitively appealing to consider the mention of an entity to evoke not only the entity itself but also other encyclopedically related entities. Many encyclopedic relationships seem to underly such an indirect evocative power: part-whole relationship, membership in a set, co-participantship in an event etc. This is supported by the fact that, many cases of *initial* references to entities involved in such relationships with already mentioned entities are *definite* NPs. The investigation of the type of textual evocation that various encyclopedic relationships can support should allow refinement of Prince's notion of textually evoked entity. It seems therefore reasonable to consider that mention of tokens of a given substance textually evokes other tokens of the same substance. Note that in EPICURE, the substance sharing relationship licenses one-anaphora only if it involves two *consecutive* tokens. In general, the textual distance between the respective mention of the evocative and the evoked tokens influences the surface form of the reference to the evoked token as much as the encyclopedic relation between the two. In EPICURE, because the current center is systematically defined as the object *of the previous clause*, whether a token is the current center or not determines whether it is considered "close" or "distant". Thus, the notion of current center plays the role of a (binary) distance subcategorization criteria for textually evoked entities.

A summary of EPICURE's reference taxonomy is given in figure 7-4. Any further subdivision inside these reference classes is based on the structure of the entity to refer to[79]. This is the case, for example, of references to brand-new entities. The microcoding of references to brand-new entities of various structures

[79] i.e. whether it is an individual, a mass, an homogeneous set or an heterogeneous set.

was presented in section 5.6.2.. In the next section, we present the microcoding of all other elements of the taxonomy.

```
      --------------------------------------------------
      |               |                                 |
 Brand-New  Containing Inferrable             Textually Evoked
                                                        |
                      ----------------------------------|
                      |                                 |
      Indirectly Textually Evoked        Directly Textually Evoked
                                                        |
                     -----------------------------------|
                     |                                  |
             Current Center                           Other
```

Figure 7-4: EPICURE's reference taxonomy

7.3.5 Microcoding non brand-new references in EPICURE

We saw in section 5.6.2 that the first question raised by microcoding a nominal referring to a GPO token is: what properties of the GPO to explicitly express in the nominal? We also saw that, in EPICURE, the answer to that question for brand new references is: express *all* properties of the GPO token. In contrast, for non brand new references, the answer is: express *a minimal set* of properties that allows the hearer to unambiguously identify the GPO referred to. Since EPICURE assumes that the hearer's discourse model is identical to its own, this means the hearer must be able to distinguish from all other GPO tokens in the discourse model. The computation of the minimal set of properties distinguishing a GPO token from all GPO tokens of the discourse model is based on the notion of **discriminatory power**. The discriminatory power *D* of a set of properties *S* of an individual or a mass is defined by the formula:

$D = (Nt - Np) / (Nt - 1)$
where: Nt = total number of GPO tokens in the discourse model
Np = number of GPO tokens in the discourse model satisfying the set of properties S.

For a set the formula is:

$D = (Nt - Np) / (Nt - C)$
where: C = is the cardinal of the set.

Recall that a token property is any attribute-value pair in the Entity Specification representing it. A set of properties distinguishes a GPO token from the GPO tokens in the discourse model if its discriminatory power is one. It is minimal if it contains less properties than any other set of properties with the same discriminatory power. Each element in EPICURE's reference taxonomy uses the notion of a minimal distinguishing set of properties in a particular way. In each of the following paragraphs we review, for a given element of this taxonomy, its particular use of this notion to choose surface form.

Reference to containing inferrable

A (containing) inferrable is a GPO token *O1* which is initially introduced in discourse indirectly by means of a reference to another GPO token *O2* whose *ancestor* feature is filled by *O1* (e.g. in *the skin of the avocado*, *the avocado* refers to a containing inferrable; before being peeled it contained *the skin*, and as a result the Entity Specification of *the skin* now includes an ancestor feature whose value is *the avocado*). References to a containing inferrable are really two references: one to the contain*ed* token and one to the contain*ing* token. Each reference is realized by an NP generated by the clause generator recursively calling itself. The NP referring to the contain*ing* token is then embedded in a PP postmodifying the head of the NP referring to the contain*ed* token (e.g. in *the kernels of a fresh ear of corn* the NP *the kernels* refers to the contain*ed* token which is an homogeneous set and the NP *a fresh ear of corn* refers to the contain*ing* token which is itself a containing inferrable).

Reference to indirectly textually evoked entity

A GPO token *T1* can be *indirectly* textually evoked if another GPO token *T2* of the same substance was mentioned in the previous sentence. In addition to substance, these two tokens can share an arbitrary number of other properties. If they share *all* other properties, then they really are two tokens of the same entity and *T1* is referred to simply by *one*. If they are distinguishable by some other properties than substance, then *T1* is referred to by an NP whose head is *one* and whose premodifiers are the minimal set of properties distinguishing *T1* from *T2* (e.g. *Slice <u>the</u> <u>green</u> <u>pepper</u>. Now remove the top of <u>the</u> <u>red</u> <u>one</u>.*).

Reference to the current center

If a GPO token is the current center of the discourse model, then there are two subcases: either it is referred to by a neutral personal pronoun (i.e. *it* if it is an individual or a mass, and *them* if it is a set) or it is altogether ellided (i.e. it is referred to by the empty string). Recall from section 5.6.2 that a GPO token always fills some participant role of an embedding eventuality token. If this participant role is nuclear, then ellipsis is ruled out and a pronoun is generated. If the participant role is satellite however, ellipsis is systematically chosen over pronominalization.

Reference to other directly textually evoked entity

First, a minimal set of properties distinguishing the GPO token from every other token in the discourse model is computed. A definite NP is then generated, with each of its constituents lexicalizing one element of this minimal set of properties. If the GPO token is either an individual, a mass or an homogeneous set, the constituents are chosen and organized inside the definite NP in the same fashion as for brand new references (cf. section 5.6.2). If it is an heterogeneous set however, instead of the conjunction of NPs built for a brand-new reference, a single definite NP is generated. The head of this NP is the most specific common superordinate of the substances of each element in the set[80] (e.g. *The vegetables* to refer an heterogeneous set of both carrots and potatoes).

We now have presented how three systems, Danlos', ANA and EPICURE, address the issue of non brand new references to object tokens. Another generator that addresses this issue is PAUL [Granville 83], presented in section 7.4.2. We compare the respective approaches of these four generators on that issue in section 7.5.2.

7.4 Cohesion

An important aspect of text generation is the study of the constraints that the textual surrounding of an utterance puts on its generation. As we saw in section 7.1, studying these constraints consists of describing the various relations binding textual units together as a unified whole. We also saw that these relations can involve either the *content* of these textual units or their *surface form*. In the first case they are said to

[80] Recall from section 5.6.2 that GPO substances are organized into a simple hierarchy.

contribute to the **coherence** of the text, whereas in the second, they are said to contribute to its **cohesion**. Because all naturally generated texts are both coherent and cohesive, only artificially uncoherent or uncohesive texts give a good feeling for what both these properties recover. Figure 7-5 gives an example of each such texts[81].

Cohesive and coherent text, *T1*:
"*The room has a large window. It faces east. It overlooks the backyard. It is located so that the sun shines through it in the morning*."

Corresponding coherent but uncohesive text, *T2*:
"*The room has a large window. The room has a window facing east. The room has a window overlooking the backyard. The room has a window through which the sun shines in the morning.*".

Cohesive but uncoherent text, *T3*:
"*Steve has a green car. His elephant likes peanuts. The car has whitewalls*."

Figure 7-5: Examples of uncohesive or uncoherent texts

Text *T3* of figure 7-5 is *uncoherent* its successive units are not related *in terms of their content*. The characteristic of coherent texts like *T1* and *T2* is that they follow a well-defined thematic progression. The necessity to enforce coherence thus constrains the answer to the question: *what to say next ?*. In other words, coherence constrains content determination and discourse organization choices. The issue of textual coherence is thus beyond the scope of the present survey.

In contrast, text *T2* is *uncohesive* because its successive units are unrelated *in terms of their interpretation*. One characteristic of cohesive texts like *T1* and *T3* is that the interpretation of some units is contingent upon the interpretation of others. This is the case of anaphoric references, such as personal pronouns or definite descriptions. Since they convey only partial information about their referent, their interpretation is contingent upon that of previous references. It is the hearer's back and forth movement between co-referent textual units during their interpretation that binds these units together. Appropriately wording anaphoric reference is thus necessary to make a text cohesive. Consequently, as opposed to coherence enforcement, cohesion enforcement constrains lexical choices. Wording references, however, is but one example of the many surface realization choices that can contribute to textual cohesion. Another example of such choice is the choice of grammatical relations between the SLUs of a text. Relations such as coordination, conjunction, subordination, embedding or relativization contribute to textual cohesion. As [Halliday 85] points out however, such relations can be used exclusively to create cohesion between units that are not only contiguous in the text but also constituents at the same linguistic level{e.g. two clauses or a clause and a nominal are at the same linguistic level, but, say a sentence and a single word are not.}. In contrast, [Halliday & Hasan 76] and [Halliday 85] discuss relations creating cohesion between units of arbitrary textual location and arbitrary linguistic levels. These relations, called **cohesive devices** result from specific surface realization choices. They are discussed in the next subsection.

[81]Due to [Granville 83].

7.4.1 Halliday's cohesive devices

Two phenomena underly the cohesive power of the devices listed in [Halliday & Hasan 76]: **co-reference** and **co-occurrence**. Two referring expressions are said to be co-referent when they both refer to the same entity. The co-reference relation between several referring expressions tie together the units these expressions are part of, thus enhancing the overall cohesion of the text. As we saw in section 7.3 there is a wide spectrum of surface forms available for making a reference. Each alternative in this spectrum is also a cohesive device. Halliday's list of co-reference related cohesive devices can be abstracted as follows:

- **Ellipsis**: Omission of (whole or part of) a constituent in an SLU (e.g. *Rod Strickland had a great game. [It was] Probably the best [game] since he is a Spur*.). Ellipsis can be considered a special case of substitution, namely substitution by the empty string. When the omitted constituent is a referring expression, the empty string realizes the reference in a totally implicit way.

- **Substitution**: Substitution of a nominal by either *one*, *some, the same* etc, e.g. *These kites are expensive, but I want one*, or substitution of a VP by *do*, *so* etc, e.g. *Who finds this girl attractive ? I do*. Resembles pronominalization in that it substitutes a constituent potentially carrying a lot of information, by a placeholder item carrying very little.

- **Pronominalization**: Reference by use of a central pronoun (e.g. *Steve had a nice wife, but he failed to keep her*) can be considered a special case of substitution.

- **Definite descriptions**: Use of a definite description for subsequent reference. Such a description can be either **abridged**, if the subsequent reference conveys less information about the referent than the initial one, or **reiterative** if it reconveys the same information.

Just as the co-presence in a text of expressions involved in a co-reference relationship makes the text cohesive, so does the co-occurrence of lexical items involved in some paradigmatic or syntagmatic lexical relations[82]. Some strong paradigmatic relations such as synonymy or hyperonymy can involve words of co-referent expressions. In this case the paradigmatic relation between the words reinforce the cohesive effect of the co-reference relation between the expressions they are part of. However, even if they appear in non co-referent expressions, the simple co-occurrence of synonyms or hyperonyms in a text enhance its cohesion. It is also true of other strong paradigmatic relations which typically exclude co-reference such as antonymy or meronymy, of weaker paradigmatic relations such as the ability to lexicalize concepts of a common encyclopedic domain and of syntagmatic relations such as collocations. Thus, the co-occurrence of words involved in any such lexical relation is a cohesive device.

Although Halliday's work provides a rather complete inventory of the lexicalization and semantic grammaticalization options available to a generator to create textual cohesion, it does not compare the respective cohesive effects of these different options. Consequently, a generator cannot use cohesion enforcement as the primary constraint on the choice among these options. Fortunately it does not have to do so. Since each of these options satisfy other constraints in different ways, these other constraints can be used to choose among these different options. For example several options for the wording of an anaphoric reference are cohesive devices. But the choice among these options is constrained by other factors such as the wording of the co-referent expression and the goals of achieving conciseness and lexical variety. If the co-referent expression is an NP with a general head noun specialized by modifiers, conciseness can be achieved by an abridged definite description where some modifiers have been dropped (e.g. *Take the block under the black pyramid with the red eye painted in the middle. Put the block on top of the stack of blocks and push the pyramid next to the Queen of Spade*.). Alternatively, lexical variety can be achieved by either

[82] see section 4.1 for the definition of various types of lexical relations.

an abridged definite description where the head noun has been replaced by one of its hyperonyms (e.g *Take half a pound of ground beef, one egg, two slices of tomatoes, some lettuce and a bun. Grill the meat* ..., or by a reiterative definite description where the head noun has been replaced by one of its synonyms (e.g. *Charles Barkley had a triple-double of 23 points, 12 rebounds and 10 assists, with half of his boards coming on the offensive end*.). In either cases cohesion is created both by co-reference and by co-occurrence of paradigmatically related words. If the co-referent expression is a proper noun, lexical variety is automatically achieved by the use of any definite description, but at the expense of conciseness.

As we saw in section 7.3.1, another constraint on the wording of reference is the status of the referent with respect to some taxonomy of Given-New information, such as that of Prince (see section 7.3.1. Appropriately marking this status in the surface form of the reference is necessary for the hearer to be able to identify the referent. Text *T2* of figure 7-5 illustrates this point. It is not only uncohesive, but more importantly ambiguous. It is unclear whether it is a single or several different windows that are successively referred to. This is an example of unappropriate marking of Given-New information by *underuse* of cohesive devices. There are also cases of unappropriate marking of Given-New information by *overuse* of cohesive devices. In such cases, for example in the second version of the story in figure 7-6, anaphoric referring expressions convey too little information for the hearer to uniquely identify the co-referents. This also leads to ambiguous texts. Therefore, there is a need in a surface generator for *controlling* the use of cohesive devices so that they create cohesion without introducing ambiguity. The task of controlling the use of several cohesive devices is the main concern of PAUL [Granville 83], the NLG system presented in the next subsection. In general, such a task involves constraint satisfaction. Constraints of appropriately marking Given-New information, constraints of achieving lexical variety, constraints of creating cohesion etc. When several of these constraints conflict it is necessary to define their relative priorities. Since it is prerequesite to unambiguity, appropriate marking of Given-New information seems the best candidate to be the overriding constraint. Conversely, since there is no available measure for the efficiency of various cohesive devices, a sound heuristic for controlling their use seems to be: maximize it within the limits set by the other constraints.

7.4.2 PAUL

[Granville 83] presents PAUL, a system that generates paragraph-length children's stories. PAUL addresses the issue of controlling the use of several cohesive devices during generation. To illustrate the nature of this issue, Figure 7-6 contains three versions of the same children story generated by PAUL from a common input. These three stories respectively result from the underuse, overuse and balanced use of the cohesive devices controlled by PAUL. Part of their common input is given in figure 7-7[83].

PAUL's input is characteristic of the type of overspecified input incorporating numerous hard-wired generation choices. This type of input allows a generator to concentrate on a very restricted class of generation choices: in PAUL's case the lexical choices involved in the microcoding of nominals referring to object tokens. PAUL's input consists in a list of event tokens, each corresponding to a proposition of the story to generate. The order in which these propositions should appear in the text is exactly the order of the elements in the input token list. This allows PAUL to ignore discourse organization choices. Moreover,

[83]We reworded this input using a feature structure notation more standard than the original NLP records [Heidorn 72] notation used by [Granville 83].

Sample story, version 1: Underuse of cohesive devices
Pogo cares for Hepzibah. Churchy likes Hepzibah too. Pogo gives a rose to Hepzibah. Which pleases her. Hepzibah doesn't want Churchy's rose. Churchy is jealous. Churchy hits Pogo. Churchy gives a rose to Hepzibah. Petals drop off. This upsets Hepzibah. Hepzibah cries.

Sample story version 2: Overuse of cohesive device
Pogo likes Hepzibah. Churchy cares for her too. He gives a rose to her. Which pleases her. She doesn't want his rose. He is jealous. He slugs him. He gives a rose to her. Petals drop off. This upsets her. She cries.

Sample story version 3: Balanced use of cohesive devices
Pogo likes Hepzibah. Churchy cares for her too. Pogo gives a rose to her. Which pleases her. She doesn't want Churchy's rose. He is jealous. He punches Pogo. He gives a rose to Hepzibah. Petals drop off. This upsets her. She cries.

Figure 7-6: Underuse, overuse and balanced use of cohesive devices

```
[((concept like)                             PFL={hepzibah,pogo}
  (grammatical-form stative-clause)
  (roles ((experiencer pogo)
          (recipient hepzibah))))
 ((concept like)                             PFL={hepzibah,churchy}
  (grammatical-form stative-clause)
  (roles ((experiencer churchy)
          (recipient hepzibah))))
 ((concept give)                             PFL={rose,hepzibah,pogo}
  (grammatical-form active-clause)
  (roles ((agent pogo)
          (affected rose)
          (recipient hepzibah)
          (effect ((concept enjoy)                     PFL={hepzibah}
                   (grammatical-form stative-clause)
                   (roles ((recipient hepzibah))))))))]
```

PAUL's common input for the first 4 clauses of the stories of figure 7-6, together with the corresponding Potential Foci Lists (PFL).

Figure 7-7: PAUL's sample input

each element in the list is annotated by a desired grammatical form[84]. This allows PAUL to ignore most semantic grammaticalization choices. PAUL just linearly scans the input list of annotated event tokens and realizes each element it encounters by a clause of the desired form. Each event concept is associated with a set of synonymous candidate verbs for its lexicalization. To realize a token of such a concept PAUL starts by *randomly* choosing the clause main verb among the set of verbs associated to that concept. This allows PAUL to achieve lexical variety without really tackling the issue of verb choice.

After having chosen the clause main verb, PAUL proceeds to realize each role of the input token. These

[84]This grammatical form specifies the voice, syntactic role structure, etc of the clause to generate.

roles are filled by object tokens of the children story microdomain[85]. These tokens are the leaves of a conceptual hierarchy where object concepts are described by a small number of static properties (i.e. properties that remains unchanged throughout the generation of the story). some of which being inherited. The conceptual hierarchy represents PAUL's encyclopedic knowledge and since PAUL assumes that the hearer's encyclopedic knowledge is identical to its own, it also represents PAUL's ESE model as a whole. Part of the conceptual hierarchy used for the generation of the stories of figure 7-6 is given in figure 7-8.

```
                              Animal
                                |
                ---------------------------------------
                |                                     |
             Mammal                                Reptile
                |                                     |
       ------------------                             |
       |                 |                            |
    Possum             Skunk                        Turtle
       |                 |                            |
     Pogo            Hepzibah                      Churchy
```

Figure 7-8: PAUL's object concept hierarchy for the generation of the stories of figure 7-6

The status of the object tokens referred to in the stories generated by PAUL with respect to Prince's taxonomy of Given-New presented in section 7.3.1 is either brand-new, indirectly textually evoked or directly textually evoked. Reference to brand-new tokens is very simply handled in PAUL: the token name is used as either a proper name, in the case of animated entities, or as the head of a simple indefinite NP in the case of unanimated entities. Reference to indirectly textually evoked tokens is also simply handled: only a single relationship is considered as support of such an indirect textual evocation, the part-whole relationship, and only within the two preceding clauses. To refer to a token which is part of another token mentioned in the these two preceding clauses PAUL systematically uses the name of the part token as a simple noun. Reference to directly textually evoked tokens, however, is performed following an algorithm controlling the use of three cohesive devices: pronominalization, reiterative definite anaphora and abridged definite anaphora. This algorithm considers a combination of encyclopedic and discursive constraints. The encyclopedic constraints are the properties the token inherits from the conceptual hierarchy, especially those distinguishing it from the other object tokens of the underlying microworld. The discursive constraints are the following:

- The textual distance of the current reference from the last reference to the same token.
- The *means* of the last reference to the same token.
- The position of the token in the **potential foci list**.

The textual distance is measured in number of clauses. The *means* of reference is the cohesive device used to word that reference. PAUL's algorithm to determine the list of potential foci is a simplification and an adaptation for generation of Sidner's focus shift tracking algorithm. It is given in figure 7-9. The potential foci list is updated for the realization of each event token in the input list. The potential foci list

[85]There are also some cases where a role of an event token is recursively filled by another event token. In such cases the clause realization procedure just described is recursively called.

for the first four clauses in the stories of figure 7-6 is given in figure 7-7 next to their respective inputs.

```
IF the desired grammatical form is an equative clause
THEN: the potential foci list has a single element, the token filling the
      subject role of the clause
ELSE: - IF the event token has an affected role
        THEN: IF this affected role is filled by an object token
              THEN: make that token the head of the potential foci list
              ELSE: IF the affected role is filled by an event token
                         with an embedded affected role
                    THEN: make the filler of the embedded affected role
                               the head of the potential foci list
      - Put the other role fillers of the event token in the potential foci
        list, with the agent role filler last
      - Put the VP lexicalizing the concept the event token is an instance
        of at the end of the potential foci list.
```

Figure 7-9: PAUL's focusing algorithm

PAUL uses encyclopedic and discursive constraints to decide between the following surface forms to refer to an object token: proper noun, personal pronoun, reiterative definite description and abridged definite description. For personal pronouns the choice is entirely dictated by the gender and number properties the token inherits from the conceptual hierarchy. For reiterative definite descriptions, the only constituent of the NP realizing the brand-new reference that is changed for subsequent co-reference is the determiner (from indefinite to definite article). In this case the cohesive effect of co-reference is reinforced by the lexical repetition of the NP heads. For definite descriptions there are two subchoices involved: choice of head and choice of premodifying noun or adjective.

The head is *randomly* chosen from the proper ancestors of the referred token in the conceptual hierarchy. Such a device ensures lexical variety. It also creates cohesion both by the loss of information between the initial[86] and subsequent reference[87] and the hyperonymy relationship between their respective heads. However, precautions have to be taken when using this device to avoid picking too general a head for the subsequent reference, which can introduce undesirable connotations (e.g. referring to a human using an NP whose head is *animal* or *thing*). To avoid such a problem, PAUL's conceptual hierarchy is cut-off at a level beyond which ancestors are not included in the list of potential head nouns for abridged definite descriptions. This cut-off is highly domain-dependent. In domains more complex than PAUL's children stories, it might be quite difficult to identify encyclopedic criteria justifying a particular cut-off. In some cases, the relations between the respective heads of two co-referring NPs introducing a given connotation might even be purely lexical.

The chosen modifiers are the properties of the referred token that allows it to be distinguished from all other descendents of the node chosen to be the head of the description. These properties can be easily determined because in the encyclopedic hierarchy representing PAUL's domain, two siblings always differ by at least one property. As we saw in section 7.3.4 this is not necessarily the case in more complex domains. In such domains, an approach based on Dale's notion of discriminatory power might be used.

[86]i.e. brand-new in Prince's terms.

[87]i.e. directly textually evoked in Prince's term.

Now that we have seen the constraints PAUL uses to choose between these three cohesive devices and the exact reference wordings resulting from each of them, we are in the position to give PAUL's algorithm to choose between these three devices. For reference to a token *T*, the algorithm can be abstracted as presented in figure 7-10

```
IF {   T is the head of the potential foci list of the current or
       preceding sentence
    OR {    T was last referred to in the two preceding clauses
        AND no other tokens with the same gender and number than T were
            last referred to in these two preceding clauses }
    OR {    T is member of the potential foci list of the preceding
            clause
        AND T was referred to in the previous sentence by mean of a
            personal pronoun filling the same role than T fills in the
            sentence currently generated }}
THEN use a personal pronoun
ELSE IF {   {    T was last referred to in the two preceding clauses
             AND no other tokens with the same gender and number than T
                 were last referred to in these two preceding clauses }
         OR {    T is a member of the potential foci list of the
                 preceding clause
             AND T was referred to in the previous sentence by another
                 mean than a personal pronoun but filling the same role than
                 T fills in the sentence currently generated }}
     THEN use an abridged definite description
     ELSE use a reiterative definite description
```

Figure 7-10: PAUL's algorithm to choose between cohesive devices

The algorithm of figure 7-10 allows PAUL to generate both cohesive and unambiguous texts in the toy-domain of children stories. However, in order to address the issue of cohesive device usage in its generality much work remains to be done. In what follow we outline some possible directions for future research on the issue:

- Using a more complete set of cohesive devices, including substitution, grammatical relations between SLUs[88](e.g. subordination, coordination, relativization) and simple co-occurrence of words involved in lexical relations orthogonal to the co-reference relation (e.g. antonymy, meronymy, collocation).
- Controlling the use of cohesive devices to satisfy other high-level goals than cohesion and unambiguity, such as informativeness, conciseness and stylistic appropriateness. Trying to satisfy such goals might require abandoning some simplifying assumptions about reference wording. For example, maximizing informativeness in a domain where a lot of information is available for each entity, requires abandoning the hypothesis that the only information to convey in a subsequent reference to an entity is the minimal information allowing the hearer to distinguish that entity from the other previously mentioned entities. Danlos' system is the only system not based on such an hypothesis, but it generates short texts, for which the issue of cohesion is not central.
- Using a hearer model to represent the possible discrepencies between the encyclopedic knowledge of the system and that of the hearer. Such a representation is necessary to determine

[88]Note that this presupposes a wider grammatical coverage and a greater flexibility in the mapping between content unit types and SLU types.

the information to be explicitly conveyed in a reference so that it is resolvable by the hearer. Since co-reference related cohesive devices are diversely located on the explicitness scale, choosing among them requires knowledge of such information.

7.5 Wording references: a comparative evaluation

We have now presented four systems addressing some aspects of the issue of generating references: Danlos' generator, ANA, EPICURE and PAUL. In this section we compare their respective approaches to that issue. Since the motivations underlying each of these solutions were quite different, this comparison is not meant to be a critical evaluation. It is rather intended as a guide to the current state of the art concerning the issue. We base this comparison on the reference taxonomy we presented in section 7.3.4. Recall from that section that the highest level distinction in this taxonomy is between references to entities that are *brand-new* to the hearer and references to entities that have already become known to the hearer in various ways[89]. Because further subcategorization is essentially based on the ontological type of the referred entity (an encyclopedic constraint) for brand-new references, rather than on the means by which the referred entity initially became known to the hearer (a discursive and/or interpersonal constraint) for *non* brand-new references, we divide this comparison in two parts, one with respect to the wording of brand-new references and the other with respect to the wording of *non* brand-new references.

7.5.1 Wording brand-new references

We base this comparison on the following criteria: linguistic coverage, encyclopedic coverage, expressive flexibility and extensibility. Let us now briefly define each of them. *Linguistic coverage* refers to the variety of SLU types used to word the references. Two levels of linguistic coverage can be distinguished:

- The *macrocoding level* which is concerned with the variety of SLU types of *whole* referring expressions (e.g. personal pronoun, indefinite NP, nominalization etc).
- The *microcoding level* which is concerned the variety of SLU types of referring expression *constituents* (e.g. for NP constituents: premodifying noun, postmodifying relative clause etc).

Encyclopedic coverage refers to the ontological variety of the domain tokens referred to (e.g. events, individual objects, masses etc). *Expressive flexibility* refers to the level of sophistication of the mapping between the domain tokens and the SLUs referring to them. Typically a domain token is a compound entity described by a set of slots. Thus, mapping a token onto an SLU can be decomposed in three stages:

1. Mapping the ontological type of the token onto a grammatical type of SLU (e.g. object token onto NP).
2. Mapping the token slots onto the possible constituents of this grammatical SLU type (e.g. *is-a* slot of the object token onto the NP head).
3. Mapping the slot values onto lexical items (e.g. value HIGH for the slot INTENSITY onto the adjective *strong* for tokens of the concept TEA, the adjective *powerful* for tokens of the concept CAR, etc).

[89]The the introduction of the concept of discursive brand-newness was mainly motivated by the desire to find an exact characterization of the discursive status of the entities for which indefinite reference is appropriate, given the many examples of initial yet definite references.

The first of these stages is constrained essentially by discursive constraints, the second by encyclopedic constraints and the third by grammatical and interlexical constraints. A distinct level of expressive flexibility can be achieved at each of these three stages:

- the *macrocoding level* at the first stage,
- the *microcoding level* at the second stage,
- the *lexical level* at the third stage.

Finally, two levels of extensibility can also be distinguished:

- The *implementation level* which is concerned with ease of extension *inside* the targeted domain.
- The *design level* which is concerned with ease of extension slightly *outside* the targeted domain.

Using these criteria, Danlos' generator, ANA, EPICURE and PAUL compare as follows:

- *Encyclopedic coverage:* All four of these systems generate references to object tokens only (i.e. they don't generate explicit references to events, processes, object properties etc.). Moreover, in the respective domains of Danlos' generator, ANA and PAUL, these object tokens are exclusively individual entities. Only EPICURE covers entities such as masses, homogeneous sets and heterogeneous sets[90]. This is one major contribution of Dale's thesis.
- *Linguistic coverage:* At the *macrocoding level*, EPICURE uses only indefinite NPs, whereas PAUL uses only proper nouns. Danlos' generator uses either one, depending on whether the token to refer to has a NAME slot. ANA's uses hybrid forms: either definite or determinerless NPs behaving much like proper nouns (e.g. *The Dow Jones average of 30 industrials*, *Wall Streets securities markets*). None of these systems use pronouns for brand-new references, because they all ignore the issue of cataphora[91]. At the NP *microcoding level*, three types of constituents are covered by EPICURE only: quantifiers, premodifying nouns and postmodifying PPs. One is covered by Danlos' generator only: postmodifying relative clauses. Both EPICURE and Danlos' generator cover premodifying adjectives.
- *Expressive flexibility:* For all four systems it is minimal at the *macrocoding level*: the mapping between token ontological types and SLU grammatical types is one-to-one. There are differences at the NP *microcoding level* however. In Danlos' generator the mapping between token slots and NP constituents is one-to-one. In EPICURE it is many-to-many and dependent on the slot *values*. At the *lexical level*, Danlos' generator, EPICURE and PAUL display no flexibility: they implement a one-to-one mapping between slot values and lexical items. Since it is macrocoded, ANA shows no flexibility at neither the microcoded level nor the lexical level.
- *Extensibility:* Because they are both made of declarative rules (production rules and rewritting rules respectively) both ANA and EPICURE are easily extendible at the *implementation level*. This is not the case in Danlos' system whose NP microcoding is carried out procedurally by a set of specialized LISP functions. At the *design level*, only EPICURE seems to be extendible.

---

[90]Note that although both Danlos' generator and ANA generates plural referring expressions, they both escape from addressing the issue of references to sets. In Danlos' generator case this is done by simply specifying the desired number of a referring expression in the corresponding input token. In ANA's case this is done by hard-wiring the plural form in the appropriate entries of the phrasal lexicon.

[91]Interpreting an abridged form of reference (e.g. pronoun, abridged definite description) requires identifying the co-referent unabridged form. In the general case, called anaphora, the unabridged form is located *before* the abridged reference. There are however, cases of *cataphora* where the unabridged from is located *after* the abridged form (e.g. *Because she was passing the record store, Judy was asked to buy Baaba Mal's new CD*).

It is based on ontological principles that grasp, albeit simplistically, some pervasive naive physics distinctions. Both ANA and Danlos' generator seem to require domains almost totally similar to those they were designed for; Danlos' generator because it uses concept-dependent *ad-hoc* heuristics to map token slots onto NP constituents; ANA because the domain entities it refers to need to be describable by a single numerical variable.

7.5.2 Wording other discursive types of reference

In the previous section we used four criteria, encyclopedic coverage, linguistic coverage, expressive flexibility and extensibility, to compare the respective approaches of Danlos' generator, ANA, EPICURE and PAUL to the wording of brand-new references. These criteria are also relevant for comparing their approaches to the wording of other discursive types of references. Moreover, for two of them, encyclopedic coverage and expressive flexibility, the resulting evaluation is identical whether for brand-new or for other types of references[92] Comparing approaches to the wording of non brand-new references, however, necessitates two additional criteria, namely cohesive coverage and discursive coverage. *Cohesive coverage* refers to the variety of cohesive devices used to word references. The cohesive devices we consider are Halliday's (cf. section 7.4.1). *Discursive coverage* refers to the variety of discursive types of the generated references. The types we consider are those of Prince's taxonomy of Given-New information (cf. section 7.3.1) and those of EPICURE's reference taxonomy (cf. section 7.3.4)[93]. Another factor for evaluating discursive coverage that we consider is the length of the co-reference chains generated. We measure this length both in terms of number of references and in terms of textual distance between the references.

Using these two additional criteria together with linguistic coverage and extensibility, Danlos' generator, ANA, EPICURE and PAUL compare as follows:

- *Linguistic coverage:* At the *macrocoding level*, ANA, EPICURE and PAUL use exclusively abridged forms of subsequent references (abridged definite description, personal pronouns). In particular, in EPICURE, the rule used partition information about the referent along a co-reference chain is to always convey *all* the information in the initial reference and to repeat a *minimal* amount of it in the subsequent ones. This approach is adequate for these systems because, due to the nature of their domain, very little information is available about each entity. However in other domains, say audit report generation, the relevant information about an entity might be rich enough so that it becomes necessary to convey it in several references. To some extent, this is what Danlos' generator does. It uses unabridged definite descriptions and attempts to equally partition the information about the referent along the co-reference chain. At the *microcoding level*, the situation is similar to that of brand-new references (cf. previous section).

- *Discursive coverage:* All four systems generate references to directly textually evoked tokens. Among these tokens, EPICURE further distinguishes between the current centre and the other tokens. Along similar lines, PAUL distinguishes between the top of the potential foci stack[94], the rest of that stack, and the other tokens. ANA, PAUL and EPICURE also generate some very restricted cases of indirectly textually evoked references. All three of them, however, consider only a single encyclopedic relationship underlying indirect textual evocation. Moreover, ANA and especially PAUL contain very few examples of such references. Only

[92]We thus do not repeat the evaluation along these criteria in this section, see previous section.

[93]Recall that EPICURE's reference taxonomy is a refinement of a part of Prince's taxonomy based on the notion of global focus space (cf. section 7.2) and current center (cf. section 7.3.2).

[94]associated with either the current or the preceding clause.

EPICURE generates references to containing inferrables. None of these systems implemented reference to other types of tokens such as non-containing inferrables[95], situationally evoked tokens and unused tokens. Finally, PAUL is the only one to consider co-reference chains longer than three clauses or containing more than three elements.

- *Cohesive coverage:* In terms of co-reference related cohesive devices all four systems use more or less sophisticated forms of pronominalization. They also all use ellipsis. However, in Danlos' system ellipsis generation is a side-effect of discourse structure choice, not a choice among several candidate cohesive devices during the lexicalization and semantic grammaticalization stage as it is in the other systems. Danlos' generator is also the only one that uses definite anaphora which are neither abridged nor reiterative but provide complementary information about the referent. ANA's and EPICURE's definite anaphoras are all abridged[96]. PAUL's can be either abridged or reiterative. Finally, EPICURE is the only system that uses a form of substitution, namely one-anaphora. In terms of co-occurrence cohesive devices, only EPICURE seems to use authentic lexical hyponymy. Its is, however, limited to subsequent references to heterogeneous sets. The form of hyponymy and synonymy ANA uses is phrasal. It is therefore a different type of relationship than the lexical level hyponymy and synonymy identified by Halliday as cohesive devices. For generating nominals, ANA performs both content determination and surface realization simultaneously as it retrieves a subject entry in its phrasal lexicon. ANA thus lacks a separate lexical level necessary to explicitly control the use of lexical co-occurrence related cohesive devices. In PAUL, such control is precluded because the correspondence between the token slot values and lexical item is one-to-one.
- *Extensibility:* At the *implementation level*, the situation is similar to that for brand-new references (cf. previous section). At the *design level*, two factors lay the ground for extension to new discursive types of reference: (1) a clear separation of the discourse model from the rest of the system, and (2) the sophistication of the current version of that model. Only EPICURE and PAUL have a separate discourse model, and that of EPICURE is by far the most elaborate. Also, only EPICURE has a hearer model to lay ground for extension to references to situationally evoked or unused tokens. It is unclear, however, that implementing any of these extensions would not require a thorough re-thinking of EPICURE's architecture.

7.6 Summary

In text generation, content units are not realized in isolation, but in the context of previous discourse. Making lexical choices sensitive to this context involves carrying out three tasks: (1) maintaining a discourse model keeping track of the underlying structure of previous discourse, (2) computing the status of incoming content units with respect to this structure, and (3) using this status as a constraint on the surface realization of these units.

In BABEL, the discursive status of a content unit is described using the notion of local focus of attention. Depending on which role of an event token is locally in focus, BABEL chooses different main verbs for the clause realizing it. Being a single sentence generator, BABEL is given focus information as input, ignoring the issue of discourse model maintenance.

In contrast, text generators like TEXT, EPICURE and PAUL all incorporates an algorithm to maintain a discourse model during generation. Also in contrast to BABEL, the type of lexical choice they constrain by

[95]Although Dale discusses the issue in his thesis.

[96]More exactly all those used for subsequent reference.

discursive factors is choice of referring nominal. TEXT maintains local focus information and uses it to decide when to pronominalize reference to object tokens. PAUL also maintains local focus information and uses it as one of several factors describing the discursive status of an object token to be referred to. The other factors include the distance from the last reference to that token, the surface form of that last reference, and the role that token fills in the embedding event token. This combination of factors determines PAUL's choice between three kinds of anaphoric references: pronominal anaphora, abridged definite anaphora and reiterative definite anaphora.

In its discourse model, EPICURE keeps track of the semantic content of all tokens referred to in previous discourse as well as the surface form of the last one. It also keeps track of the current center of ongoing discourse, a notion very similar to that of local focus. It uses this information to computes the status of an object token to be referred to, with respect to a refinement of Prince's subcategorization of Given-New information. This status, together with the ontological type of the token in EPICURE's complex underlying domain, determines the choice between four types of anaphoric references: ellipsis, one-anaphora, pronominal anaphora and various forms of abridged definite anaphoras. Both in terms of the sophistication of its DSE model and in terms of the variety of lexical decisions it constrains upon this model, EPICURE is by far the most interesting system.

Anaphoric reference is one of two linguistic phenomenon underlying textual cohesion, the other being lexical co-occurrence. Uncohesive texts are not only clumsy and artificial but also sometimes ambiguous. Thus, text generators which are not maintaining an explicit discourse model, such as Danlos' system or ANA, incorporate some practical mechanism controlling the use anaphoric references.

8. Conclusion

To conclude our survey we give a list of limitations of the lexicalization and semantic grammaticalization tasks implemented by state-of-the-art NLG systems. Limitations on lexicalization can be grouped in four broad categories: limitations on the kinds of semantic content units accepted as input, limitations on the kinds of SLUs specified in output, limitations on the kinds of mapping implemented and limitations on the kinds of knowledge sources used. Each of the following sections briefly discusses one of these four types of limitations and suggests some possible directions for future research.

8.1 Limitations in terms of input

In terms of input to lexicalization, one limitation is on the complexity of the ontological modelling of the underlying domain. All existing generators distinguish between two top-level ontological categories: objects and events. However, domain models incorporating a mechanism allowing dual views[97] on entities have yet to be developed. It is also unclear that everything one needs to represent as a stand-alone entity in any given domain naturally falls into either element of this dichotomy (e.g. general properties of physical objects such as color, texture etc). Moreover, the ontological spectrum of both object and events represented in existing generators is also quite limited. For objects, that spectrum is generally restricted to individual objects. EPICURE's modelling of masses and of various kinds of sets is a first step toward a greater variety of input object tokens. Along the same line, one unsolved issue of pervasive importance is the modelling of spatial and functional decomposition of compound objects. For events, the ontological spectrum is generally restricted to atomic actions and their composition using state-variable based planning. But as pointed out by many authors (e.g. [Allen & Kautz 85], [Kowalsky & Sergot 86], [Moens & Steedman 87], [Passoneau 87]), actions are but one restricted category of events and state-variable based planning does not allow the representation of most kinds of compound event decomposition, in particular of those grammatically marked by aspect and tense. The representation of causality is also a major issue for future investigation. These issues of domain ontology are unavoidable as generation moves toward more complex underlying domains. They are extremely hard, however, since they bring along naive physics and commonsense reasoning which might be the current bottleneck of knowledge representation research (see [Hobbs & Moore 85]).

The other limitation on a generator input is how much semantic and language independent it is. The input of many existing systems, implicitly contains some lexicalization choices. The exception is ANA, which numeric input is more remote from linguistic structure than the generation-oriented conceptual structures used in most other systems. However, ANA's domain is also exceptionally simple. Starting from more language independent input in increasingly complex domains will require to tackle the lexicalization task to its full extent. However, striving for more semantic and language independent input should not become a self-contained (and thus somewhat artificial) goal to pursue like in BABEL. To yield interesting results, it must be motivated by a practical application in a well-circumscribed domain like in ANA.

[97] i.e. one object view and one event view.

8.2 Limitations in terms of mapping

In human-generated texts, content units of a given ontological type are realized by diverse types of SLUs. Events are an example of such content units: although they are typically realized by clauses (e.g. *The Lakers beat the Bulls twice this year*), they can also be realized by NPs (e.g. *The two victories of the Lakers against the Bulls this year*.). NPs realizing events are often called **nominalizations** in the literature. Object properties are another example of content units whose possible realizations cut across SLU categories. They can be realized by adjectival phrases (e.g. *Western Conference top-seeded Portland*), relative clauses (e.g. *Portland which is the top-seeded team in the West*), non-finite clauses (e.g. *Portland top-seeded in the West*), appositive NPs (e.g. *Portland, the top-seeded team of the Western Conference*) etc. In existing generators, however, the mapping between content unit types and SLU types in invariably one-to-one at the macrocoding level (event tokens onto clauses and object tokens onto nominals) and generally also one-to-one at the microcoding level[98]. This seriously limits their expressive flexibility. In particular, since different SLU types allow different degrees of satisfaction of high-level generation goals, such as informativeness, conciseness, readability, cohesion, stylistic appropriateness, being able to choose among several of them to express each content unit is necessary to maximize the satisfaction of such goals.

In human-generated texts a given content unit type can be realized not only by SLUs of different *types* but also different *sizes*. For example an object property can be lexicalized by simple insertion of an adjective in the NP referring to the object (e.g. *With his steady shooting and outstanding defense job on Michael Jordan, Joe Dumars led the Pistons to a 86-77 win against the Bulls*.) but also by the juxtaposition of an entire equative sentence (e.g. *Joe Dumars' steady shooting and defense led the Pistons to a 86-77 win against the Chicago Bulls. His defense job on Michael Jordan was outstanding*.). Thus, abandoning a one-to-one mapping between content unit *types* and SLU *types* also requires abandoning a one-to-one mapping between content unit *size* and SLU *size*, as implemented by most existing generators. Engineering a many-to-many mapping between content unit size and SLU size, would allow choosing between explicit and implicit expression of semantic relations between content units. It also raises the issue of the interaction between lexicalization and low-level discourse organization. Except for Danlos', very little work has been devoted so far to the investigation of this issue.

8.3 Limitations in terms of output

When evaluating the linguistic coverage of a generator it is important to distinguish among the various SLU types appearing in the texts, those whose members are either in one-to-one mapping with content units or randomly chosen, from those which are actually (i.e. non-randomly) chosen in context at generation time. Only the second are authentic cases of lexical choices. Also among these, it is important to distinguish between SLU types whose members are simply retrieved as a whole from the lexicon (i.e. macrocoded) from those which are constructed from lower-level ones[99] (i.e. microcoded). This is because macrocoding allows achieving wide linguistic coverage at the expense of expressive flexibility.

With respect to linguistic coverage, there are two main categories of generators, those performing actual choices of several SLU types at the macrocoding level and those which microcode a single SLU type but does not perform any actual choices for the other SLU types. The prototypical example of the first category

[98]see section 5.6.5 for the precise definition of these mapping levels.

[99]These lower-level ones were being in turn either macrocoded or recursively microcoded.

is ANA which performs actual choices of both clause patterns and nominals. The prototypical example of the second category is EPICURE which microcodes nominals in detail but maps event tokens onto clause patterns in a one-to-one fashion. Danlos' generator falls somewhat in the middle, choosing less hard-wired clause patterns than those of ANA and performing some minimal form of nominal microcoding as well. But the texts generated by Danlos' system are only three clauses long, as opposed to the two to three paragraphs texts generated by ANA and EPICURE. Thus, no existing system is yet able to fully microcode both clauses and nominals in multi-paragraph text generation. Moreover, no significant attempt has been made yet to perform actual choice for either the most complex forms of clauses and NPs nor for other SLU types such as PPs, adjectival phrases, adverbials phrases, genitives, partitives etc. Another limitation of existing systems concerns their coverage in terms of semantic subtypes inside grammatical SLU types. For example, most systems generate only action clauses, thus ignoring equative and attributive clauses. As generation moves toward new and more complex domains, attempts to perform non-random macrocoded choices and then to microcode an increasing number of SLU types will have to be made. We feel, however, that improving the linguistic coverage and the level of macrocoding will not result in any spectacular increase in the quality of computer-generated texts before content unit types are mapped onto SLU types in a many-to-many fashion. Finally, concerning the issue of microcoding, it is important to recall that it considerably increases the complexity of both the generator and the lexicon. Thus, fully microcoded text generation with wide linguistic coverage in significantly complex domains is a long-term goal. As suggested by [Kukich 83], a first step toward such goal is the building of generators able to switch between several microcoding gears depending on SLU types. This would allow concentrating the research effort on the macrocoding of some SLU types while macrocoding the others.

8.4 Limitations in terms of knowledge sources

Concerning the knowledge sources it uses to perform lexicalization, a generator can be limited in essentially three ways:

- In terms of the diversity of constraints encoded in these knowledge sources.
- In terms of the way these constraints are encoded in these knowledge sources.
- In terms of the way these knowledge sources were acquired.

In chapter 2 we identified five types of constraints on lexical choice: grammatical constraints, interlexical constraints, encyclopedic constraints, interpersonal constraints and discursive constraints. In each of the following chapters we saw that each of these constraint types itself recovers many different constraints that should be encoded by the corresponding knowledge sources. In each existing system, however, at most two out of these five types of constraints has been used to perform lexicalization *at a significant scale*: in Meaning-Text Theory based report generators grammatical and interlexical constraints, in Danlos' system grammatical and encyclopedic constraints, in BABEL, PHRED and ANA only encyclopedic constraints, in EPICURE encyclopedic and discursive constraints, in PAULINE only interpersonal constraints and in PAUL only discursive constraints. Attempting to build a generator representing a significant number of all five types of constraints and using them for lexicalization is a challenging direction for future research. We feel that this could be successfully pursued only by a coordinated effort of a large team of researchers focused on a single, practical application. It would however shed much light on the controversial but crucial issues of representation and control of multiple constraints.

The representation of multiple constraints can be either integrated or modular. It can also be either procedural or declarative. For example, in Danlos' generator encyclopedic and discursive constraints are integrated and procedurally represented in the LISP functions in charge of nominal microcoding whereas, encyclopedic and grammatical constraints are integrated and declaratively represented in the lexicon-grammar. In EPICURE discursive and encyclopedic constraints are modularly and declaratively represented in the rewriting rules of the clause generator. Given that knowledge sources for lexicalization are bound to become large entities developed by several people over significantly long periods and at times using a trial and error approach, using declarative representations for these sources makes good sense. The advantage of integrated representations is that they allow for less sophisticated generation architectures by relieving the generator from the burden of handling constraint interaction. This interaction is implicitly encoded by hand at knowledge acquisition time in the integrated knowledge structures. In contrast modular representations where each type of constraints in separately represented, requires architectures able to combine these separate constraints at generation time. For such architectures to remain simple, these separate constraints should ideally be encoded using a single formalism. Modular architectures significantly ease the knowledge acquisition process[100] in two ways:

- By allowing the knowledge engineer to concentrate on a single aspect of the overall lexical choice issue.
- By laying ground for the use of existing knowledge sources.

This is especially important because, the potential for portability across applications is very uneven among the different types of constraints on lexicalization. For example, the portability potential of grammatical and interlexical constraints in quite high, whereas that of encyclopedic constraints is extremely limited. Thus, integrating, say, grammatical and encyclopedic constraints should be avoided.

Knowledge acquisition will undoubtedly constitute a major issue for future research in NLG. In this field, almost everything remains to be done: all existing systems were developed both from scratch and without the help of any knowledge acquisition tool. Concerning the use of portable constraint sources, the best bet for the immediate future are grammatical and interlexical constraints. NIGEL is the first effort to develop a portable wide-coverage grammar. The last decade has seen considerable effort dedicated to the building lexical databases from machine readable dictionaries (see [Amsler 80], [Calzolari 84], [Alshawi 87], [Boguraev & Briscoe 87] among others) none of these databases has yet been used by an NLG application to constrain lexical choice.

One reason reason for this is the inadequacy of the information contained in dictionaries for computational purposes in general and for generation in particular. Having being written for human usage, they lack the formal rigor and systematicity needed by computers. Moreover, dictionaries are indexed by words while a generation lexicon needs to be indexed by concepts.

Another reason for this state of affair is the absence of any standardization of generation technique. As it clearly appears from the NLG systems we surveyed, no two generators have much in common, whether it be in terms of types of lexical choices performed, in terms of types of constraints encoded, in terms of internal representation formalism or in terms of architecture. Therefore it is almost impossible to develop a knowledge source readily usable by several generators whose designs differ so much. For example, we just noted that a generation lexicon needs to be indexed by concepts. But for such a lexicon to be used by several generators, these generators must share the same representation formalism for concepts. We believe this abscence of standardization is also the main reason behind the current lack of knowledge acquisition and software engineering environments for NLG systems. NLG systems tend to become large software

[100] which, for NLG like for any knowledge intensive task constitutes the bottleneck of the development effort.

systems which development require such environments. However, such environments are themselves large systems, so the cost of building them is economical only if they are used for the development of several generators within a given framework. We feel that before a better consensus is reached concerning NLG systems architecture, the development of such environments is unlikely to be economical. However, it is an interesting research issue on its own, that some researchers started to investigate: [Nirenburg & Raskin 87] discusses methodologies to build knowledge acquisition systems for NLP applications and [Smadja 89] shows how large textual corpora can constitute an alternative to machine-readable dictionaries as source of lexical knowledge source for NLG systems.

8.5 Summary

The lexicalization task consists of mapping content units onto grammatical specifications of surface linguistic units with all open-class lexical items chosen. In the future this mapping will be many-to-many and take as input an increasing ontological variety of content units. All types of output surface linguistic units will be non-randomly chosen among several alternatives and an increasing number of them will be macrocoded. This will enhance the expressive flexibility, linguistic coverage and encyclopedic coverage of NLG systems. An increasing variety of constraints will be used in concert to perform lexicalization. In order to achieve portability across applications for some types of constraints, such as grammatical and interlexical constraints, different constraints will have to be represented declaratively and independently, and combined at generation time. Software environments will be developed to speed up the critical knowledge acquisition phase.

References

[Allen 87] Allen, J.F.
Natural Language Understanding.
Benjamin/Cummings, Menlo Park, CA, 1987.

[Allen & Kautz 85]
Allen, J.F., Kautz, H.A.
A model of naive temporal reasoning.
Formal theories of the commonsense world.
Ablex Publishing Corp., 1985.

[Allen & Perrault 80]
Allen, J.F. & Perrault, C.R.
Analysing intentions in utterances.
Artificial Intelligence 15(1):143-178, 1980.

[Alshawi 87] Alshway, H.
Processing dictionary definitions with phrasal pattern hierarchies.
Computational Linguistics 13(3-4):195-202, March, 1987.

[Amsler 80] Amsler, R.A.
The structure of the Merriam-Webster Pocket Dictionary.
PhD thesis, University of Texas, 1980.

[Appelt 83] Appelt, D.
TELEGRAM: a grammar formalism for language planning.
In *Proceedings of the 8th International Joint Conference on Artificial Intelligence*. IJCAI, 1983.

[Appelt 85] Appelt D.
Studies in Natural Language Processing: *Planning Natural Language Utterances.*
Cambridge University Press, 1985.

[Bach 86] Bach, E.
The algebra of events.
Linguistics and philosophy (9):5-16, 1986.

[Barwise & Perry 83]
Barwise J. & Perry J.
Situations and attitudes.
MIT Press, 1983.

[Benson, Benson & Ilson 86]
Benson, M., Benson, E. & Ilson, R.
The BBI combinatory dictionary of English: a guide to word combinations.
John Benjamins Publishing Company, Amsterdam/Philadelphia, 1986.

[Bobrow & Winograd 77]
Bobrow D. G., & Winograd T.
An overview of KRL, a knowledge representation language.
Cognitive science 4(1), 1977.

[Boguraev & Briscoe 87]
Boguraev, B. & Briscoe T.
Large lexicons for natural language processing: utilising the grammar coding system of LDOCE.
Computational Linguistics 13(3-4):203-218, March, 1987.

[Boyer & Lapalme 85]
Boyer, M. & Lapalme, G.
Generating paraphrases from meaning-text semantic networks.
Computational Intelligence (3&4), August-November, 1985.

[Brachman 79] Brachman, R. J.
On the epistemological status of semantic networks.
Associative networks.
Academic Press, New York, 1979.

[Brennan, Friedman & Pollard 87]
Brennan, S.E., Friedman, M.W. & Pollard, C.J.
A centering approach to pronouns.
In *Proceedings of the 25th Conference of the ACL*, pages 155-162. ACL, 1987.

[Calzolari 84] Calzolari, N.
Machine-readable dictionaries, lexical databases and the lexical system.
In *Proceedings of the 10th International Conference on Computational Linguistics*. COLING, 1984.

[Carcagno & Iordanskaja 89]
Carcagno, D. & Iordanskaja, L.
Content determination and text structuring in GOSSIP.
1989
Presented at the 2nd European Generation Workshop.

[Chafe 76] Chafe, W.L.
Giveness, contrastiveness, definiteness, subjects, topics and points of view.
Subject and Topic.
Academic Press, New York, 1976.

[Charniak 77] Charniak, E.
Ms. Malaprop, a language comprehension program.
In *Proceedings of the 5th International Joint Conference on Artificial Intelligence*. IJCAI, 1977.

[Choueka, Klein & Newitz 83]
Choueka, Y., Klein, T. & Newitz, E.
Automatic retrieval of frequent idiomatic and collocational expressions in a large corpus.
ALLC journal 4:34-38, 1983.

[Church 90] Church, K.
Word Association Norms, Mutual Information and Lexicography.
Computational Linguistics 16(1):22-29, March, 1990.

[Cumming & Albano 86]
Cumming, S. & Albano, R.
A guide to lexical acquisition in the Janus system.
Technical Report ISI/RR-85-162, ISI, Marina del Rey, CA, 1986.

[Dale 88] Dale, R.
Generating referring expressions in a domain of objects and processes.
PhD thesis, University of Edinburgh, 1988.

[Danlos 86] Danlos, L.
Studies in Natural Language Processing: *The linguistic basis of text generation.*
Cambridge University Press, 1986.

[Danlos, Guez & Sabbagh 85]
Danlos, L., Guez, S. & Sabbagh, S.
Systemes d'aide: vous avez dit intelligent ?
In *Proceedings of COGNITIVA 85*. COGNITIVA, 1985.

[Debili 82] Debili, F.
Analyse syntaxico-semantique fondee sur une acquisition automatique de relations lexicales-semantiques.
PhD thesis, University of Paris 6, 1982.
These de doctorat d'etat.

[Ducrot 84] Ducrot, O.
Propositions: *Le dire et le dit.*
Les editions de minuit, Paris, 1984.

[Elhadad 87] Elhadad, M.
The link between situation and language use: toward a more coherent interaction.
Technical Report CU-CS-289-87, Columbia University, 1987.

[Elhadad 88] Elhadad, M.
The FUF Functional Unifier: User's manual.
Technical Report CUCS-????, Columbia University, 1988.

[Elhadad 90] Elhadad, M.
Constraint-based text generation: using local constraints and argumentation to generate a turn in a conversation.
Technical Report CU-CS-????, Columbia University, 1990.
Thesis proposal.

[Fikes & Nilsson 71]
Fikes, R. E. & Nilsson, N. J.
STRIPS: a new approach to the application of theorem proving to problem solving.
Artificial Intelligence (2), 1971.

[Freckelton 84] Freckelton, P.
Une taxonomie des expressions idiomatiques anglaises.
PhD thesis, Universite de Paris 7, 1984.

[Goldman 75] Goldman, N.
Conceptual Generation.
Conceptual Information Processing.
North-Holland, Amsterdam, 1975.

[Granville 83] Granville, R.
Cohesion in Computer Text Generation: Lexical Substitution.
Technical Report MIT/LCS/TR-310, MIT, 1983.

[Gross 75] Gross, M.
Methodes en syntaxe.
Hermann, Paris, 1975.

[Gross 84] Gross, M.
Lexicon-Grammar and the analysis of French.
In *Proceedings of the 11th International Conference on Computational Linguistics*. COLING, 1984.

[Gross 86] Gross, M.
Lexicon-Grammar: The representation of compound words.
In *Proceedings of the 11th International Conference on Computational Linguistics*. COLING, 1986.

[Grosz 78] Grosz, B.J.
Focusing and description in natural language dialogues.
1978
Presented at the Workshop on Computational Aspects of Linguistic Structure and Discourse Setting, University of Pennsylvania, May 24-27, 1978.

[Grosz & Sidner 86]
Grosz, B. & Sidner, C.L.
Attentions, intentions, and the structure of discourse.
Computational Linguistics 12(3):175-204, 1986.

[Grosz, Joshi & Weinstein 81]
Grosz, B.J., Joshi, A.K. & Weinstein, S.
Providing a unified account of definite noun phrases in discourse.
In *Proceedings of the 21st Conference of the ACL*, pages 44-50. ACL, 1981.

[Halliday 76] Halliday, M.A.K.
System and function in language.
Oxford University Press, Oxford, 1976.

[Halliday 85] Halliday, M.A.K.
An introduction to functional grammar.
Edward Arnold, London, 1985.

[Halliday 68] Halliday, M.A.K.
Notes on transitivity and theme in English; Parts 1,2 & 3.
Journal of linguistics (3.1, 3.2 & 4.2), 1967-68.

[Halliday & Hasan 76]
Halliday M.A.K. & Hasan, R.
Cohesion in English.
Longman Group Limited, London, 1976.

[Heidorn 72] Heidorn, G.E.
Natural language inputs to a simulation programming system.
Technical Report NPS-55HD72101A, Naval Postgraduate School, Monterey, CA, 1972.

[Hobbs & Moore 85]
Hobbs, J.R, & Moore, R.C. (editor).
Formal theories of the commonsense world.
Ablex Publishing Corp., 1985.

[Hovy 87] Hovy, E.
Generating natural language under pragmatic constraints.
PhD thesis, Yale University, 1987.

[Jacobs 85] Jacobs, P.
PHRED: a generator for natural language interfaces.
Computational Linguistics 11(4), 1985.

[Kameyama 86] Kameyama, M.
A property-sharing constraint in centering.
In *Proceedings of the 24th Conference of the ACL*, pages 200-206. ACL, 1986.

[Kay 79] Kay, M.
Functional Grammar.
In *Proceedings of the 5th Annual Meeting of the Berkeley Linguistic Society*. 1979.

[Kittredge & Mel'cuk 83]
Kittredge, R. & Mel'cuk, I.A.
Towards a Computable Model of Meaning-Text Relations within a Sublanguage.
In *Proceedings of the 8th International Joint Conference on Artificial Intelligence*. IJCAI, 1983.

[Kittredge, Polguere & Goldberg 86]
Kittredge, R., Polguere, A. & Goldberg, E.
Synthesizing weather forecasts from formatted data.
In *Proceedings of the 11th International Conference on Computational Linguistics*. COLING, 1986.

[Kowalsky & Sergot 86]
Kowalsky, R. & Sergot, M.
A logic-based calculus of events.
New Generation Computing (4), 1986.

[Kukich 83] Kukich, K.
Knowledge-based report generation: a knowledge engineering approach to natural language report generation.
PhD thesis, University of Pittsburgh, 1983.

[Leed & Nakhimovsky 79]
Leed, R.L. & Nakhimovsky, A.D.
Lexical functions and language learning.
Slavic and East European Journal 23(1), 1979.

[Lyons 77] Lyons, J.
Semantics.
Cambridge University Press, 1977.

[Maarek 90] Maarek, Y.
Automatically Generating Natural Language Help Systems From Free-Style Textual Documentation.
In Jacobs, P. and Croft, B. and Waltz, D. (editor), *Proceedings of the AAAI Symposium on Text Based Intelligent Systems*, pages 66-70. AAAI, Stanford, Ca, March, 1990.

[Mac Gregor & Bates 87]
Mac Gregor, R. & Bates, R.
The Loom knowledge representation language.
Technical Report ISI/RR-87-188, ISI, 1987.

[Mann 82] Mann, W.C.
An overview of the PENMAN text generation system.
Technical Report ISI/RR-83-114, ISI, 1982.

[Mann 87] Mann, W.C.
Text generation: the problem of text structure.
Technical Report ISI/RR-87-181, ISI, 1987.

[Mann & Matthiessen 83]
Mann, W.C. & Matthiessen, C.M.
NIGEL: a systemic grammar for text generation.
Technical Report ISI/RR-83-105, ISI, 1983.

[Mann & Thompson 87]
Mann, W.C. & Thompson, S.
Rhetorical Structure Theory: description and constructions of text structures.
Natural Language Generation: New Results in Artificial Intellligence, Psychology and Linguistics.
Martinus Ninjhoff Publishers, 1987.

[Matthiessen 87] Matthiessen, C.M.
The organization of the environment of a text-generation grammar.
Natural Language Generation: New Results in Artificial Intellligence, Psychology and Linguistics.
Martinus Ninjhoff Publishers, 1987.

[Mayo 61] Mayo, B.
Objects, events and complementarity.
Philosophical review (LXX):340-361, 1961.

[McCardell 88] McCardell, R.
Lexical selection for natural language generation.
1988
PhD. Thesis Proposal, Computer Science Department, University of Maryland.

[McCoy 86] McCoy, K.F.
The ROMPER system: responding to object-related misconceptions using perspective.
In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*. ACL, 1986.

[McDonald 80] McDonald, D.
Natural Language Generation as a Process of Decision-Making under Constraints.
PhD thesis, MIT, 1980.

[McDonald 83] McDonald, D.
Description Directed Control: its Implications for Natural Language Generation.
Readings in Natural Language Processing.
Morgan Kaufmann Publishers, 1983.
also in Computers & Mathematics #9(1).

[McDonald, Vaughan & Pustejovski 87]
McDonald, D.,Vaughan, M. & Pustejovski, J.
Factors contributing to efficiency in natural language generation.
Natural Language Generation: New Results in Artificial Intellligence, Psychology and Linguistics.
Martinus Ninjhoff Publishers, 1987.

[McKeown 85a] McKeown, K. R.
The need for text generation.
Technical Report CUCS-173-85, Columbia University, 1985.

[McKeown 85b] McKeown, K. R.
Studies in Natural Language Processing: *Using Discourse Strategies and Focus Constraints to Generate Natural Language Text.*
Cambridge University Press, 1985.

[McKeown & al. 90]
McKeown, K. R., Elhadad, M., Fukumoto, Y., Lim, J.G., Lombardi, C., Robin, J. & Smadja, F.A.
Text generation in COMET.
Current Research in Natural Language Generation.
Academic Press, 1990.

[McKeown & Swartout 87]
McKeown, K.R. & Swartout W.R.
Language generation and explanation.
The Annual Review of Computer Science (2):401-449, 1987.

[Mel'cuk 81] Mel'cuk, I.A.
Meaning-Text Models: a Recent Trend in Soviet Linguitics.
The Annual Review of Anthropology (10), 1981.

[Mel'cuk 82] Mel'cuk, I.A.
Lexical functions and lexicographic description.
In *Proceedings of the 8th annual meeting of the Berkeley linguistic society*, pages 427-444. University of California Berkeley, 1982.

[Mel'cuk & al. 84]
Mel'cuk, I.A. & al.
Recherches lexico-semantiques: *Le Dictionaire Explicatif et Combinatoire du Francais contemporain.*
Presses de l'Universite de Montreal, Montreal, 1984.

[Mel'cuk & Pertsov 87]
Mel'cuk, I.A. & Pertsov, N.V.
Surface-syntax of English, a formal model in the Meaning-Text Theory.
Benjamins, Amsterdam/Philadelphia, 1987.

[Mel'cuk & Polguere 87]
Mel'cuk, I.A. & Polguere, A.
A formal lexicon in the Meaning-Text Theory.
Computational Linguistics 13(3-4):261-275, 1987.

[Minsky 75] Minsky, M.
A framework for representing knowledge.
The psychology of computer vision.
McGraw-Hill, New York, 1975.

[Moens & Steedman 87]
Moens, M. & Steedman, M.
Temporal ontology in natural language.
In *Proceedings of the 25th Conference of the ACL*, pages 1-7. ACL, 1987.

[Moore 80] Moore, R.C.
Reasoning about knowledge and action.
Technical Report 191, SRI International AI Center, Menlo Park, CA, 1980.

[Ney 83] Ney, J.W.
Optionality and choice in the selction of order of adjectives in English.
General Linguistics (23):94-128, 1983.

[Nirenburg 87] Nirenburg, S. (editor).
Studies in Natural Language Processing: *Machine Translation: theoretical and methodological issues.*
Cambridge University Press, 1987.

[Nirenburg 89] Nirenburg, S.
Lexical selection in a blackboard-based generation system.
1989
forthcoming.

[Nirenburg & Raskin 87]
Nirenburg, S., Raskin, V.
The subworld concept lexicon and the lexicon management system.
Computational Linguistics 13(3-4):276-289, 1987.

[Paris 85] Paris, C.
Towards more graceful interaction: a survey of question-answering programs.
Technical Report CUCS-209-85, Columbia University, 1985.

[Passoneau 87] Passoneau, R.J.
Situations and intervals.
In *Proceedings of the 25th Conference of the ACL*, pages 16-24. ACL, 1987.

[PENMAN 88] The PENMAN Natural Language Generation Group.
The PENMAN user guide.
1988
Draft.

[Prince 81] Prince, E.
A taxonomy of Given-New information.
Radical pragmatics.
Academic Press, New York, 1981.

[Queneau 58] Queneau, R.
Exercises in style.
New directions publishing corporation, New York, NY, 1958.
Translated from French by Barbara Wright.

[Quirk & al. 72] Quirk, R. & al.
A grammar of contemporary English.
Longman, 1972.

[Raccah 87] Raccah, P.Y.
Modelling argumentation, modelling with argumentation.
1987
to appear in the journal ’Argumentation’.

[Reichman 85] Reichman, R.
Getting computers to talk like you and me: discourse context, focus and semantics (an ATN model).
MIT press, Cambridge, MA, 1985.

[Reiter 90] Reiter, E.
Generating descriptions that exploit a user’s domain knowledge.
1990
presented at the 1st European Generation Workshop.

[Rodale & al. 47] Rodale, J.J. & staff.
The word finder.
Rodale Books Inc., Emmaus, PA, 1947.

[Sacerdoti 77] Sacerdoti, R.
A structure for plans and behaviors.
North-Holland, Amsterdam, 1977.

[Salkoff 83] Salkoff, M.
Bees are swarming in the garden.
Language 59(2), 1983.

[Saussure 16] Saussure, F. de.
A course in general linguistics.
Philosophical Library, New York, 1916.

[Schank 75] Schank, R. (editor).
Conceptual Information Processing.
North-Holland, Amsterdam, 1975.

[Schank 82] Schank, R.
Dynamic Memory: a Theory of Reminding and Learning in Computers and People.
Cambridge University Press, 1982.

[Schank & Abelson 77]
Schank, R.C. & Abelson, R.P.
Scripts, plans, goals and understanding.
Lawrence Erlbaum Associates, Hillsdale, NJ, 1977.

[Searle 69] Searle, J.
Speech acts: an essay in the philosophy of language.
Cambridge University Press, Cambridge, 1969.

[Searle & Vanderveken 85]
Searle, J. & Vanderveken, D.
Foundations of Illocutionary Logic.
Cambridge University Press, Cambridge, 1985.

[Sidner 79] Sidner, C.L.
Towards a computational theory of definite anaphora comprehension in English discourse.
Technical Report AI-TR-537, Computer Science Department, MIT, Cambridge, MA, 1979.

[Simmons & Slocum 72]
Simmons, R. & Slocum, J.
Generating English discourse from semantic networks.
Communication of the ACM 15(10), 1972.

[Smadja 89] Smadja, F.A.Z.
Generating collocationally restricted sentences using automatically acquired co-occuremce knowledge.
1989
PhD. Thesis Proposal, Computer Science Department, Columbia University.

[Watt 88] Watt, M.
The realization of natural language with pragmatic effects.
Technical Report CSRI-215, Computer Systems Research Institute, Univerity of Toronto, 1988.

[Wilensky & al. 84]
Wilensky, R., Arens, Y. & Chin, D.
Talking to UNIX in English: An overview of UC.
Communications of the Association for Computing Machinery 27(6), 1984.

[Winograd 83] Winograd, T.
Language as a cognitive process.
Addison & Wesley, 1983.

[Woods 70] Woods, W.A.
Transition network grammars for natural language analysis.
Readings in Natural Language Processing.
Morgan Kaufmann Publishers, 1970.
also in CACM 3(10).

Table of Contents

List of Figures