Time-Constrained Memory For Reader-Based Text Comprehension

Jean-Pierre Corriveau

Technical Report CSRI-246
April 1991

Computer Systems Research Institute
University of Toronto
Toronto, Canada
M5S 1A1

The Computer Systems Research Institute (CSRI) is an interdisciplinary group formed to conduct research and development relevant to computer systems and their application. It is an Institute within the Faculty of Applied Science and Engineering, and the Faculty of Arts and Science, at the University of Toronto, and is supported in part by the Natural Sciences and Engineering Research Council of Canada.

TIME-CONSTRAINED MEMORY FOR READER-BASED TEXT COMPREHENSION

by

Jean-Pierre Corriveau

A Thesis submitted in conformity with the requirements for the Degree of Doctor of Philosophy in the University of Toronto

Abstract

Marvin Minsky (1986, p.18) writes at the beginning of *The Society of Mind* that "to explain the mind, we have to show how minds are built from mindless stuff, from parts that are much smaller and simpler than anything we'd consider smart." In this dissertation, I develop a model of a strictly quantitative (*i.e.,* non-semantic) memory that can be used to specify a conceptual analyzer for *teuchistic* (*i.e.,* 'constructionist') text comprehension. I view this model as a prototype of Minsky's "agents of the mind".

Most importantly, I acknowledge the real-time processing constraints derived from the *biological constraint* (Feldman, 1984) and therefore, assume that linguistic comprehension is a race defined in terms of *time-constrained* memory processes.

Because I do not model an adaptable memory, I partition memory into a static component, which consists of a massively parallel network of simple computing elements whose processes allow for the construction of clusters, and a dynamic component, where these clusters reside. Through specification browsers, the user of the system can input and modify both the topology of the network and the individual behavior of each computing element of static memory, which forms a 'knowledge' base. Clusters are built from the processing of an input text with respect to this 'knowledge' base and constitute the output of the system. Given that there is widespread disagreement on the nature, *modus operandi*, and use of inferences in text comprehension, the focus in this work is not on the knowledge required for comprehension, but rather on its specification in terms of constraints to satisfy through the exchange of simple signals and sequences of primitive memory operations to execute upon constraint satisfaction. I demonstrate at length how typical rules for the problems of syntax, referential resolution, lexical and structural disambiguation, and bridging inferences can be encoded in the proposed representational scheme, and thus illustrate how a theory of text understanding may be 'grounded' into a more fundamental quantitative time-constrained

memory.

Acknowledgements

I would like to thank first my advisor, Graeme Hirst, who initiated and nurtured my interest in computational linguistics and cognitive science. His patience and guidance throughout my research have transformed a chaotic collection of ideas into this dissertation.

I am also grateful to my external examiners, David Waltz and James Hendler, as well as to the members of my thesis committee: Hector Levesque, John Mylopoulos, Stephen Regoczei, and Ken Sevcik.

I am indebted to Edwin Plantinga for the countless discussions that helped me conceive, simplify, and criticize my ideas, and to Nicola Santoro who provided crucial support. I would also like to thank Raymond Aubin, George Berg, Michel Corriveau, Judy Dick, Tom Fairgrieve, Ken Forsythe, Helen Gigley, Diane Horton, Susan McRoy, and Stephen Regoczei for their help.

I am grateful to my parents, Roland and Madeleine, for giving me the opportunity to study. I cannot thank them enough for their constant encouragement, support, and love. In the end, this work would never have been completed if it wasn't for the loving presence of my wife, Valérie Chaplain, who always believed in me more than I did.

Financial support from the Natural Sciences and Engineering Research Council of Canada and from Bell-Northern Research is gratefully acknowledged.

Contents

Part I

A Model of Time-Constrained Memory

Chapter 1

Rationale

1.1 Introduction

Since the early 1970s, significant advances have been realized in the field of natural language processing (NLP). But research has mainly focused on user interfaces and the parsing of isolated sentences; the processing of larger linguistic units has typically remained a stumbling block (Habel, 1983; Winograd and Flores, 1986, chapter 9). More specifically, there are currently few computational models that tackle the *understanding* (or equivalently, *comprehension*) of long, unrestricted, *written text* (henceforth, text). I use the term 'text' in the broad sense in which Charles Muller (1977, p.5, Martin Phillips's translation) defines it as:

> any utterance or any succession of utterances, any use of speech or fragment of speech, with no restriction on its extent, produced by a single speaker or writer and displaying a certain unity.

This "certain unity" of text is taken to be central to comprehension: *subject matter* (or equivalently, '*aboutness*') is what gives a text this certain unity. It is generally accepted that if we fail to perceive the subject matter of a text, we find it difficult, if not impossible, to understand that text (Bransford and Johnson, 1973). In this dissertation, I address the problem of text comprehension. and more specifically. the issue of the *perception* of subject matter. I view the *expression*, as opposed to the perception, of subject matter as a distinct problem, which has more to do with the tasks of language generation and memory recall (Baddeley, 1976; Kintsch and van Dijk, 1978; van der Meer, 1987), which are not addressed in this research.

Of the computational models that have been proposed over the last eighteen years for text comprehension (*e.g.,* Schank, 1972, 1982; Cullingford, 1978; Dyer, 1983; Wilensky, 1978, 1983b), most are conceptual analyzers primarily concerned with the problem of inference; they adopt what Peter Norvig (1989) calls "a strong method", introducing complex symbolic representational mechanisms and schema-matching sequential algorithms to recognize specific classes of inferences. The disadvantage of these models is that new algorithms and data structures have to be created every time a knowledge structure is proposed. Moreover, it is commonly recognized that widespread disagreement exists about the nature, classification, and generation mechanisms of knowledge structures for inference in text understanding (Graesser and Clark, 1985, p.1).

Rejecting methods tuned to process a particular set of knowledge structures, other researchers have instead proposed conceptual analyzers that replace schema-matching by *marker-passing*[1] (*e.g.,* Hirst, 1987; Stallard, 1987; Charniak and Goldman, 1988; Hobbs, Stickel, Martin, and Edwards, 1988; Pollack and Pereira, 1988). In particular, Norvig (1987, 1989) presents a model of text comprehension based on a marker-passing algorithm that detects six general classes of inferences. Marker-passing systems typically combine a parallel semantic network with a inherently sequential path evaluator: through the propagation of markers, paths corresponding to inferences (Charniak, 1986b) are constructed in the semantic network and submitted to the path evaluator. These systems typically suffer from both (1) the computational inefficiency of either having to sequentially consider a large, if not unmanageable, number of inferences suggested by their semantic network, or to depend on an intractable evaluation process (*e.g.,* abduction), and (2) the very little consideration given to their representational and algorithmic complexity.

In contrast with the relative representational and processing arbitrariness of conceptual analyzers, the connectionist paradigm proceeds from the computational constraints imposed by neuronal modeling (Feldman, 1984, 1985a, 1985b). In the last five years, considerable

[1] Given a semantic representation of the next input, markers are passed from each concept of this representation to adjacent nodes, following the links of the semantic network specified by the knowledge base. Markers start out with a given amount of *marker activation* or *energy*, and are spread through the network, spawning new markers with less energy, stopping when the energy value reaches a minimum. Markers are information structures from which the *path* to the original concept, that is, the concept that initiated the passing, can be requested. When two or more markers are passed to the same node, a *marker collision* occurs. For each such collision, the associated path consists of one half originating from the initial representation, and one half that leads to the inferred node. Each collision denotes a *possible* inference. Therefore, the path associated with each collision must be evaluated according to some fixed *a priori* criteria that cause the inference to be either made, rejected, or even deferred.

effort has been expended in order to develop NLP systems built on the massively parallel architectures and distributed processing that characterize connectionism. The resulting models (*e.g.,* Cottrell, 1984, 1985; Selman, 1985; Waltz and Pollack; 1985; McClelland and Kawamoto, 1986; Selman and Hirst, 1987) are typically single-sentence parsers that produce a pattern of activation corresponding to a parse tree. Some of these models can tackle simple lexical and structural disambiguation, or very simple inferences (*e.g.,* Eiselt and Granger, 1987). With the notable exception of McClelland and Kawamoto's work, these models typically adhere to the 'local connectionism' paradigm (*i.e.,* 'one node equals one concept') that is distinct from the dominant parallel distributed processing (PDP) flavor of connectionism (McClelland and Rumelhart, 1986). The fundamental feature of conceptual analyzers, especially with respect to text comprehension, is that, contrary to neuronal models and, in particular, to connectionist ones, they use symbolic data structures and algorithms to *build* a representation of the input.

Most recently, a few researchers have attempted to integrate marker passing and local connectionism by presenting hybrid systems that employ some elements of connectionism within a marker-passing mechanism: *e.g.,* link weights, activation values and thresholds (Chun and Mimo, 1987; Lange, Hodges, Fuenmayor, and Belyaev, 1989; Lee, Flowers, and Dyer, 1989), and microfeatures (Hendler, 1989). These models, however, are not directly concerned with the computational constraints of neuronal modeling, and still generally rely on complex symbolic data and processes (*e.g.,* unique concept identifiers (*e.g.,* Lange and Dyer, 1989), complex markers, variable binding) and heuristics (*e.g.,* rules of composition for paths). Closer to 'pure' connectionism is George Berg's proposal (1987) of an enhanced parallel semantic network with nodes that have the ability to change links and add new nodes to the network. Nodes of the network are activated by the energy spreading through the underlying network eliminating the need for a centralized controller. Spreading activation is also the basis for a simple conceptual analyzer.

The research presented in this dissertation constitutes another attempt at integrating conceptual analysis with the computational constraints imposed by neuronal modeling. Unlike previous research, I do not target my work to a particular cognitive task (*e.g.,* schema selection) nor to a specific implementation problem (*e.g.,* the integration of connectionism and marker passing, or Berg's goal of obtaining a network with distributed control), but rather construct the model from a more theoretical viewpoint, as will be explained below.

1.2 Human Memory as Basic Metaphor

The apparent dichotomy between conceptual analyzers and neuronal models can be viewed as an instance of the well-known and ubiquitous mind-body dilemma (Johnson, 1987). I wish to bypass this dilemma by respecting the Aristotelean inseparability of these two entities. Acknowledging that one or more 'basic metaphors' (Mac Cormac, 1985) underlies any theory of knowledge representation and processing, I therefore choose to root the cognitive architecture I propose in the basic metaphor of human memory that is taken to embody this inseparability.[2] Yet, human memory constitutes a vast open interdisciplinary research field in which controversies and contradictions abound (Baddeley, 1976, 1986; Tulving, 1983, 1984; Arbib, Conklin, and Hill, 1987), and it is not my intent to develop a model that accounts for the latest evidence in all of the numerous relevant disciplines. In other words, human memory[3] as a basic metaphor is too broad and must be further restricted within a computational framework. This is my goal for the rest of this section.

The fundamental hypothesis of neuronal modeling, a basic metaphor rooted in the body, is that if we acknowledge that the human brain is involved in the act of comprehension, then the consideration of its anatomy and physiology may provide helpful insights for the design of a computational model of cognition (Feldman, 1984, 1985a, 1985b). All neuronal models proceed from the so-called 'biological constraint' (*ibid.*) that consists of the following observations about the brain:

- Neurons are orders of magnitude slower than current electronic devices.
- Neurons exchange 'simple' non-symbolic signals.
- Neurons carry out 'simple' computations.

From these observations, neuronal researchers conclude that memory consists of a large network of simple computing elements working in parallel and having properties that cannot be captured at a 'higher' symbolic level. Conversely, proponents of conceptual analysis, a basic metaphor rooted in the mind, typically ignore the biological constraint and parallelism, that is, deny the importance of the 'lower' neuronal level, to develop instead arbitrarily complex knowledge structures and algorithms that correspond to arbitrarily powerful processing

[2] Human memory is undeniably biological (Squire, 1987), but also acts as the medium for the storage and processing of elusive mental entities such as 'ideas', 'beliefs', 'dreams', etc.

[3] I use hereafter the word 'memory' to refer to human memory as opposed to electronic storage.

units engaging in arbitrarily complex communication. The perspective I take in this thesis is that of the fundamental hypothesis of neuronal modeling: I assume that the biological constraint imposes important processing constraints on cognition as a whole and therefore, I adopt the consequent conceptualization of memory as a massively parallel network of simple computing units exchanging simple signals. Memory processes are constructed on top of this network, and are taken to underlie the processes involved in cognitive tasks such as linguistic comprehension.

Conceptual and local-connectionist models are programmable, that is, their behavior is controlled by the designer. In conceptual models, programmability is achieved through the specification of data structures and algorithms. In 'localist' models, through the direct modification of the weights of the connections of the network that, within the connectionist paradigm, encode all knowledge in the network. Conversely, parallel distributed processing (PDP) models, once trained, are adaptable, that is, their behavior is controlled by the network itself, which directly adjusts the weights of its connections. Conrad (1985) observes that there is an inherent tradeoff between programmability and adaptability. In this dissertation, it is not my intent to develop a theory of human learning, but rather to construct a *text comprehension tool* based on a model of memory that respects the biological constraint. Therefore, I will not focus on the mechanisms that shape human memory into the remarkable adaptable system it is, and I will not address the problems associated with adaptable systems (*e.g.*, learning, chunking, proceduralization, categorization). Instead, in accordance with the reader-based strategy that will be introduced shortly, the proposed tool will be highly programmable. More specifically, the *user* of the proposed tool will be able to define and modify the major processing parameters and 'knowledge' (*i.e.*, qualitative as opposed to quantitative data) used by this tool to generate an interpretation of an input text. Also, just as connnectionist models use only highly idealized 'neurons', I will not present a neuronal model of memory *per se*, that is, a model that accounts for all anatomical, physiological, psychological, chemical, and neuropsychopharmacological evidence (among the multitude of relevant fields).

The conceptualization of memory as a massively parallel network of simple computing units exchanging simple signals says little about the *modus operandi* of memory. In general, neuronal models are relaxation models (see Selman, 1985; McClelland and Rumelhart, 1986): once inputs have been entered in the network, the computation terminates when the network reaches an equilibrium that is mathematically defined, typically as a global function to minimize (Feldman, 1985b). In other words, there are no rules to define the completeness of the computational process(es) used to solve a cognitive task or rules to specify the correctness of the solution reached (Rumelhart, 1984): a solution emerges from a constraint satisfaction process rather than from a search through a space of solutions. In contrast, conceptual approaches rely on data structures, rules, heuristics, and algorithms that must cover every possible input, and thus define in effect the completeness of the understanding algorithm and the 'correctness' of the solution obtained.

Within the framework of linguistic comprehension, I feel close to Lindsay and Manaster-Ramer's (1987) *teuchistic* approach to cognitive processes:

> Our view is that [natural language] use involves processes which are neither algorithmic nor heuristic but, to coin a new term, **teuchistic**. Teuchistic processes construct a solution rather than search for one.... We discard the concepts of *language* and *grammar*, replace the customary notion of linguistic competence with a concept of *potential behavior*, and view production and comprehension as nonmonotonic problem-solving processes that satisfice rather than search out an optimal choice from a pre-established theoretical set of possibilities. We further suggest that the multi-component constraint satisfaction model serve as the basis of such problem solving. Comprehension is modelled as a constraint driven satisficing process: information is selectively attended and used to constrain the generation of an interpretation.

These authors present several general patterns of linguistic processing by human beings supporting a teuchistic model. In particular, they remark that:

> people produce and parse rapidly a vanishingly small proportion of the theoretically possible utterances of a given language, and moreover, intend and understand only a vanishly small proportion of the theoretically possible interpretations of those utterances. Yet given sufficient time and effort, people seem to be able to increase these proportions significantly.

Lindsay and Manaster-Ramer explicitly reject the concept of a 'correct' or 'optimal' interpretation. The notion of a single determinate correct meaning to be retrieved from a text can also be abandoned on hermeneutic grounds, as will be argued subsequently.

In this dissertation, I take a position that follows from the viewpoint of Lindsay and Manaster-Ramer. In particular, I adopt a *reader-based* approach to text understanding, one in which 'meaning' is constituted by the interaction between text and reader (Holub, 1984). The reader is not seen as an idealized competent entity, but rather as an individual (with all the idiosyncrasies implied by this term) that comes to a personal interpretation of a text with respect to a private idiolect. Therefore, the concept of linguistic *competence* (which is inaccessible by definition) is abandoned at both the sentential and textual levels of interpretation, as will be explained subsequently. Thus no claims are to be made about 'linguistic knowledge' or about the comprehension model itself; we focus only on actual *performance*, which is taken to be idiosyncratic. It follows that the only object of scientific study in my work is the suggested *computational* model of memory, not the 'knowledge' it manipulates (which is taken to be idiosyncratic): all qualitative rules (*e.g.*, rules of syntax, semantic definitions of ambiguous words, schemata, etc.) presented in this dissertation will be strictly illustrative and no claim will be made with respect to ill-defined metrics such as the 'correctness' or 'normality' of these rules.[4] In other words, I focus on the form, rather than on the content, of the 'knowledge' required for text comprehension. Throughout the dissertation I will put the word 'knowledge' between quotes in order to emphasize that the user of the proposed comprehension tool specifies all qualitative data necessary for understanding. This includes 'rules' that extend beyond the symbolic rules of conceptual analyzers.

I abandon the notion of a correct interpretation of a text and thus, I forsake the quest for rules, algorithms, and heuristics—a standpoint shared by connectionism, teuchistic models, and reader-based approaches to linguistic comprehension. Yet memory, which is taken to underlie linguistic comprehension, *must* follow some functional principle corresponding to an algorithm. Charles von der Malsburg (1985) solves this dilemma when he remarks that:

> - There is a very clear division between the qualitative and the quantitative aspects of an algorithm.
> - With the abandonment of algorithmic control, the brain can be hypothesized to follow a strictly quantitative 'trivial algorithm' that fixes only the general form of operations; all qualitative information (*e.g.*, rules, concepts, etc.) is treated as data.

It follows that within the framework of my basic metaphor:

- All memory processes are taken to be strictly quantitative *i.e.*, mechanical and deprived of any linguistic and semantic knowledge.
- All 'knowledge', that is, all qualitative information, manipulated by the proposed comprehension tool is assumed to be strictly user-specifiable.

[4] Interestingly, I have noticed that, as a native speaker of French and a second-language user of English, my rules of parsing, especially for prepositional phrase attachment, can differ considerably from those learned or idiosyncratically constructed by native English speakers. Other informants informally confirm this observation.

- The processes of linguistic comprehension must be defined only in terms of quantitative memory processes that manipulate user-specified qualitative data.

In other words, the goal of this research is *not* to develop rules and algorithms for comprehension, but, rather, to present a quantitative model of memory 'on top' of which a conceptual analyzer can be specified.

Lindsay and Manaster-Ramer suggest that linguistic comprehension consists in a teuchistic (*i.e.*, construction) process constrained by:

> a variety of factors, only some of which bear a close resemblance to the linguistic knowledge as normally conceived, whereas others involve factors such as the past processing that has been done by the system, interaction with other users of the language, and even physical characteristics of the language user.

Of the multitude of possible factors that constrain linguistic comprehension, I focus primarily in this work on the role and importance of quantitative *time*, that is, time as it pertains to memory management and memory processes such as retrievals, (see Corriveau, 1987). More specifically, the fundamental and most pervasive hypothesis of this dissertation is that, generally, linguistic comprehension is a *time-constrained process*—a *race*. The idiosyncrasies of interpretation observable between readers stem, in part, from differences between the private time constraints of these readers, that is, from differences in the comprehension time allocated by each one. This hypothesis proceeds not only from the biological constraint that emphasizes the short response times of humans for hard cognitive tasks (Feldman, 1985a; Gigley, 1985a, 1985b), but also from psychological evidence suggesting that humans feel a determinative pressure to understand quickly (*e.g.*, Márkus, 1983, for inference; Norris, 1986, for word sense disambiguation; van Dijk and Kintsch, 1983, for text comprehension), possibly as the result of social demands and struggles (Peckham, 1979). From this viewpoint, the hypothesis considers both body and mind, in accordance with the postulate about the inseparability of these two entities.

The idea of linguistic comprehension as a time-constrained process eliminates the need to choose between relaxation mechanisms and arbitrary rules to stop the understanding process. Instead, time becomes the essential stopping criterion: there is no correct interpretation, but rather an interpretation that is reached given a certain private knowledge base and a set of time-related memory parameters that characterize the "frame of mind" (Gardner, 1983) or "horizon" (Gadamer, 1976) of a particular individual. The 'acceptability' of this interpretation, a metric that could be captured through a global multi-variable function to minimize (much like relaxation models), is not addressed in this work for it depends on a multitude of factors (*e.g.,* social criteria) that lie beyond the scope of my basic metaphor.

At this point of the discussion, let me recapitulate my main working hypotheses and limitations:

- Memory is a massively parallel network of simple computing units that exchange simple signals.

- Mechanisms that shape human memory into an adaptable system will not be addressed.

- The notion of a single determinate correct meaning to be retrieved from a text is abandoned in favor of a reader-based approach to text understanding.

- Memory follows a ***trivial algorithm***, that is, a strictly quantitative algorithm that fixes only the general form of memory operations. All memory processes are strictly quantitative.

- All 'knowledge', that is, all qualitative information manipulated by the proposed comprehension tool, is strictly user-specifiable.

- Linguistic comprehension is a teuchistic process that must be defined in terms of quantitative memory processes.

- Of the multitude of possible factors that constrain linguistic comprehension and its underlying processes, the focus is on the importance of quantitative time.

- The notion of the acceptability of interpretation will not be addressed.

Given these assumptions, the task at hand is to develop a model of memory, that takes into account the computational constraints imposed by neuronal modeling and can be used for conceptual analysis. Such a model, called **IDIoT** (for Idiosyncratically Directed Interpretation of Text), has been developed. Let me briefly summarize it. Through specification tools, the user of **IDIoT** can input, browse, and modify both the topology of the memory network and the individual behavior of each computing element in memory. The network constitutes the repository for all qualitative data, that is, the 'knowledge' base that the tool uses during the interpretation of a text. The comprehension process consists of the construction of cognitive structures in memory. These structures represent the output of **IDIoT** and can be inspected by the user. In order to initiate an interpretation by **IDIoT**, the user selects a 'knowledge' base and an input text. Each word of the text activates some computing element(s) of the network (*e.g.*, word 'meanings', parsing rules, inferences, etc.). Upon its activation, each network element may send signals to other network elements, construct new cognitive structures in memory, or modify the existing ones at that point in time. Except for the network elements directly activated by the words of the text, all computing elements of memory are triggered by certain signals received from other memory elements and require a small amount of time to become activated. (Details are provided in chapter 3.)

At this point of the dissertation, I must emphasize that to assume that the biological constraint imposes computational constraints on cognition is not to claim the biological plausibility of the proposed model of memory. Indeed, such a claim would seem to inherently presuppose a solution, which we do not have, to the mind-body problem. In the same vein, I remark that this thesis is not concerned with the representational debate between information-processing and connectionism, nor with semantic considerations such as the correspondence between the semantic network specified by the user of **IDIoT** and formal semantics (*e.g.*, default logic, Etherington and Reiter, 1985). The object of study of my work is the model of quantitative time-constrained memory and I do not focus on the characteristics of the possible semantic representations that a user of **IDIoT** might adopt. In other words, the representational level of the proposed memory is 'lower', more quantitative (*i.e.*, mindless), than typical 'semantic' networks. The model distinguishes itself from connectionism in both its programmability and its consideration of processing time. From my viewpoint, **IDIoT** constitutes a prototype of Marvin Minsky's (1986, p.18) "agents of the mind":

> To explain the mind, we have to show how minds are built from mindless stuff, from parts that are much smaller and simpler than anything we'd consider smart. Unless we can explain the mind in terms of things that have no thoughts or feelings of their own, we'll only have gone around in a circle. But what could those simpler particles be—the "agents" that compose our minds? There are many questions to answer.... These questions all seem difficult, indeed, when we sever each one's connection to the other ones. But once we see the mind as a society of agents, each answer will illuminate the rest.

This dissertation illustrates how mind (and its semantic rules, which form a conceptual analyzer) can emerge from a society of mindless agents, that is, more specifically, from the interactions of strictly quantitative memory processes.

1.3 Overview of the Dissertation

In part 1 of this thesis, I develop the proposed model of memory.

Chapter 2 encompasses the theoretical foundations of the work. It begins with a review of the currently dominant position in the cognitive sciences and of the existing approaches to text understanding, and then argues that the information-processing paradigm rests on a *structuralist* and *objectivist* theory of interpretation in which text is seen as the repository of a single determinate meaning, placed in it by the writer; a 'competent' reader merely retrieves this meaning. My contention is that this is too restrictive an approach and that the notion of 'readerly competence' leads to a normative and self-validating theory of comprehension that merely begs the question. From this standpoint, I abandon attempts at specifying a correct set of rules of comprehension and, instead, argue for a reader-based approach to text understanding. Within the framework of my basic metaphor, this choice is shown to constrain the design of an algorithm for linguistic comprehension, that is, the specification of a *formal* a priori *computation*, to the quantitative aspects of memory. The research relating to **IDIoT** is discussed throughout this chapter.

In chapter 3, I develop a model of time-constrained memory from the working hypotheses presented above. First, memory is partitioned into a user-specified static component and a dynamic, temporally organized, component that holds the cognitive structures (called *clusters*) constructed by the teuchistic process of comprehension. The internal structure and behavior of the elements of static memory, called *knowledge units* (KUs), is then presented. In essence, a KU acts as a feature detector and as a cluster builder. A feature is a qualitative entity and its detection consists of satisfying a local constraint of a KU. The qualitative data of the model is not in the connections of the network, but rather in the KUs, which are small strictly quantitative finite state machines implementing forward and backward chaining. Each KU contains an *expansion procedure*, which is a sequence of primitive cluster operations. Once a KU satisfies a local constraint, its expansion procedure is executed by the *memory manager*, which is responsible for the manipulation and management of clusters. Through the execution of its expansion procedure, a KU can modify the contents of the dynamic memory, allowing for both the enforcement and application of the rule(s) associated with the detected feature. The chapter proceeds with a discussion of the implementation of the model. I then present an annotated example. To conclude, I summarize the different programmable facets of the proposed model and discuss some of the insights gained from experiments with the current prototype.

In part 2, I address the specification of a conceptual analyzer over time-constrained memory. In chapter 4, I first review three prototypical models of text understanding from which the different tasks of linguistic comprehension are identified. Then, in the next five chapters, for each of the tasks of syntactic processing, reference resolution, word-sense disambiguation, structural disambiguation, and bridging inferences, I argue for the omnipresent importance of time-constrained memory and demonstrate how a user of the proposed tool can specify rules that capture simplified solutions to these problems. I must emphasize again that the goal is not to specify 'correct' rules, but to illustrate how typical rules can be encoded in **IDIoT**. Several of the 89 examples running with the current prototype will be presented in scenarios that consist of sequences of events summarizing the actual sequences of messages exchanged.

Finally, the last chapter summarizes the main goals, achievements, and limitations of this work. The dissertation closes with a brief discussion of several possible enhancements to the proposed model.

1.4 Terminology and Conventions

For convenience, I will list here some of the terms used throughout this work:

- **Cluster:** An element of dynamic memory. A cluster holds a set of features, each feature governing a set of clusters.
- **Feature:** A qualitative entity detectable by one or more knowledge units.
- **Knowledge base** (KB): The repository of all qualitative data defined by the user.
- **Knowledge unit** (KU): A user-defined element of static memory, that is, of the KB.
- **Short Term Memory** (STM): Short-term memory in the conventional psychological sense.

A few conventional abbreviations are also employed for syntactic categories:

- **NP**: noun phrase.
- **PP**: prepositional phrase.
- **VP**: verb phrase

In a grammar rule, a superscript asterisk on an item means that the item may be repeated zero or more times; square brackets denote optionality.

The name of a feature appears in bold. Names follow the Smalltalk-80 (Goldberg, 1984) convention: they may be arbitrarily long and consist of a single word in which some letters may be capitalized to simplify reading.

Chapter 2

Models of Understanding

2.1 The Conduit Metaphor

According to the *Webster's Ninth New Collegiate Dictionary*[1], "*language* is a systematic means of communicating ideas or feelings by the use of conventionalized signs, sounds, gestures, or marks". Martin Phillips (1985, p.3) observes:

> There is one ultimate fact about text. This is that it consists of elements of linguistic substance juxtaposed in linear sequence. In the case of written text, the ...reader somehow internalises from the encounter with graphic substance ... the meaning of the text.

For written text, the marks (or graphemes) recorded on a certain medium (*e.g.*, paper) are physical (graphical) instantiations of linguistic elements. These linguistic elements do not have physical substance *per se*: they belong to the reader's *mind*, that is, the entity or set of entities that controls abilities such as understanding, feeling, perceiving, thinking, willing, and reasoning. For example, the letter *t* does not exist in a three-dimensional blot of ink on a piece of paper, but rather in the mind of a cognitive agent. By looking at the blot of ink, the cognitive agent may *perceive* (or equivalently, *apprehend*) a *t*, or may fail to do so.

The linguistic elements that form a text have a symbolic nature: they *symbolize* other *mental* (or equivalently, *cognitive*) entities typically called *ideas* or *meanings*. It is generally accepted that the perception of the subject matter of a text involves the perception of the meaning of the constituents of the text. In other words, it is typically assumed that the perception of the subject matter of a text requires the ability to perceive the meaning of

[1] Merriam-Webster Inc., 1981, Springfield, MA, p.641.

the smaller linguistic elements that form the text. For simplicity, let us appeal momentarily to the intuitive notions of *word*, *phrase*, and *sentence* to refer to these smaller linguistic elements.

Many researchers have attempted to explain the relation of language to meaning. Michael Reddy (1979) has suggested that a complex metaphor, which he calls the *conduit metaphor*, underlies most current linguistic theories. This metaphor disposes us to think of linguistic communication as follows:

- Ideas are mental objects.
- Linguistic expressions are containers.
- Communication is sending.

To effect communication, a speaker puts ideas *into* words and then sends them to a hearer who takes the ideas *out* of the words. What a linguistic expression *means* depends solely on what the speaker put into the container (Plantinga, 1986).

For written text, the conduit metaphor is writer-based; the reader merely retrieves the *determinate* meaning that the author put in the sentences. John Searle (1979, pp.117–119) has summarized the most common view of meaning that proceeds from this metaphor:

> Sentences have literal meanings. The literal meaning of a sentence is entirely determined by the meaning of its component words (or morphemes) and the syntactical rules according to which these elements are combined. A sentence may have more than one literal meaning (ambiguity) or its literal meaning may be defective or uninterpretable (nonsense).... The literal meaning of the sentence is the meaning it has independently of any context whatever; and diachronic changes apart, it keeps that meaning in any context in which it is uttered.

Assuming that each word in language has a few possible meanings that are readily accessible greatly simplifies the task of sentence understanding: the meaning of a sentence is determined by the meaning of the words that form it. Words are taken to *refer* to 'reality' and therefore the meaning of an utterance can be obtained by evaluating the *correspondence* of the meaning contained in its words to reality. With the 'correct' algorithm, 'the' meaning of a sentence is obtained. Similarly, it is hoped that with the 'correct' algorithm, 'the' meaning of a text will be *computable* from the meaning of the words and sentences of the text. To put this another way, the linguistic elements that form the text carry their own meaning, they constitute *information*, and the reader receives meanings contained in words. According to this viewpoint, the reader uses an *information*-processing algorithm: by recognizing the *rules of composition* of meaning used by the author, the reader is able to retrieve the subject matter of the text. Meaning and subject matter are objectified through the words of the text, comprehension is *normalized* with respect to the set of rules of composition used by the writer. As Edwin Plantinga (1987) observes, "the individual has been banished from contemporary linguistics...linguistics studies language but not *homo loquens*".

This concern with normalized understanding by a *competent* comprehender, one who possesses and uses the correct set of rules, can be traced back to Ferdinand de Saussure (1916), who was the first of the few modern linguists who has achieved fundamental insight into the problem of meaning. The first principle of his *Cours de Linguistique* emphasizes the *arbitrariness* of the linguistic *sign*: the linguistic sign consists of an arbitrary relationship between a *signifiant* (or symbol) and a *signifié* (or referent). In other words, Saussure asserts that there is no necessary correspondence between the systems of language and our experience of phenomena. But if the relationship is arbitrary, it must also be *conventional* "for if the relationship between 'signifiant' and 'signifié' were not conventional, there could be no question of exercising stylistic choice or of creativity of the poetic work which attaches new and unexpected meanings to familiar words" (Phillips, 1985, p.7).

Phillips remarks that the crucial consequence of the principle is that signs are not mutually substitutable and that, therefore, a system has to be established to keep them distinct in use. Both extreme homonymy (*i.e.*, multiple referents mapping into identical signs) and extreme synonymy (*i.e.*, a single referent realized by a multitude of signs) would lead not to an *unworkable* system, but to the *absence* of system. Thus, with the acceptance of the fundamental Saussurian tenet, the rest of linguistics can be seen as a specification of the limits to arbitrariness.

The information-processing paradigm, I repeat, focuses on the conventions of linguistic comprehension and ignores the arbitrariness of linguistic communication. The paradigmatic goal is to find *the systems* of language and meaning, that is, sets of rules for understanding. Since the relation of language to meaning is highly complex, several separate classes of systems have been proposed. Let me review these different classes and discuss their adequacy for text comprehension.

2.2 Information Processing and Text Comprehension

2.2.1 On Story Grammars and Discourse Analysis

The definition of *language* quoted above suggests that a certain systematicity is generally accepted for linguistic communication. Typically, language is viewed as consisting of a number of interrelated systems operating at different levels of analysis. The commonly recognized levels of linguistic analysis are the phonological, the graphological, the morphological, the lexical, the syntactic, the semantic, the pragmatic, and the discourse level. For the comprehension of written text, the phonological level of analysis does not enter into consideration. "Nor does the graphological one since typographical features do not operate at a unique level of linguistic analysis, but rather cut across a number of different levels. Moreover, although such features make some contributions to the overall semantics of the text, it is a relatively superficial one. The morphological level of analysis is also inappropriate to the investigation of text[:] word morphology relates to the function of words in syntactic frames; that is, it largely reflects syntactic structuring." (Phillips, 1985, p.30–32).

Within the information-processing paradigm, it is generally agreed that the perception of subject matter depends on "global semantic structures of text-as-whole", known in psychology as *macrostructures* (van Dijk, 1980):

> We should distinguish between the (general) *form* of a narrative and its (actual) *content*, which is of course an old insight.... We will call a *story* any discourse which has a narrative structure. Hence, we distinguish between a *discourse type* (stories), its *narrative global form* (superstructure[)], its narrative global content (macrostructures; which may be conventionalized[)] and, of course, the actual linguistic expression of these in the form of a sequence of sentences: a discourse.

Phillips (*ibid.*, pp.3–4) explains:

> It seems, then, that appreciation of textual meaning is a large-scale phenomenon which does not depend directly on those structures which are responsible only for the local organisation of linguistic expressions.... It is widely accepted that ...non-linear conceptual structures are elaborated by the reader and are the mechanism which underlies the reader's ability to summarise, paraphrase and generally state what the text is about. This ability raises some interesting problems. It has been pointed out that to be able to state what a book is about depends on the processing of thousands of sentences which cannot normally be memorised individually by the reader.... In general, it is the 'gist' of a text which is recalled rather than the wording.

In modern theoretical linguistics, syntactic concerns occupy the center of attention in investigations of *grammar* (*i.e.*, the set of rules required to assemble meaningful symbols into a meaningful utterance). In particular, the theory of *generative grammars*, whose most notable proponent is Noam Chomsky (1965, 1980, 1982), introduces powerful notational mechanisms that can be carried over to the textual level. It has therefore been suggested that the meaning of a text can in some sense be accounted for within the framework of grammatical theory.

Story grammars were initially proposed to reformulate Vladimir Propp's (1968) theory of Russian folktales. David Rumelhart (1975) suggested the first more general grammar. This grammar's first rewrite rule was:

Rule 1: *story → setting + episode*

This rule states that a story is composed of an element called a 'setting' followed by an element called an 'episode'. Each of these components is defined by subsequent rules: a setting, as a sequence of stative propositions, and an episode, as an event followed by some reaction of the protagonist of this episode. Each rewrite rule is associated with a semantic constraint. For example, for the rule above, the semantic constraint specifies that the setting must 'enable' the episode.

Some researchers (*e.g.*, Johnson and Mandler, 1980) have presented more elaborate grammars in which *transformations* specify the passage from the *surface structure* of a story to its *base* (or *deep*) structure. It is generally assumed among story grammarians (the notable exceptions being Nancy Johnson and Jean Mandler) that the proposed grammars characterize cognitive *schemata* used during comprehension (van Dijk, 1980). The realization of these schemata in language is of no import to these researchers (see comments of Allen in Wilensky, 1983a).

A very critical review of the story grammar paradigm can be found in Robert Wilensky's (1983a) article "Story Grammars versus Story Points" and the comments of his peers that follow the paper. Allan Garnham (1983) also vigorously argues against the approach. The immediate difficulty with a story grammar is that it ultimately relies on the intuitions of the grammarian himself for the definition and recognition of its *terminals* (*e.g.*, 'event'). Furthermore, the relevance of the grammatical framework to text comprehension is rather dubious: perception of subject matter is not an issue of grammaticality, that is, of well-formedness, but of what a text is about.

From my point of view, discourse analysis (see de Beaugrande, 1980, for reviews) is very similar to story grammars in that it focuses on the techniques used by the writer to introduce the topic, rather than on the perception of the subject matter of a text itself. Though these techniques are relevant to understanding, *how* something is presented is not sufficient to account for *what* is presented. Phillips (1985, p.35) elaborates:

> I am left with the uncomfortable feeling that not only is the propositional content of text relevant to [its understanding], but is the central issue and one which in discourse analysis is necessarily avoided.... Moreover, the highest unit of analysis at present generally recognised within discourse analysis is the 'event'.... The unit is, however, not very well defined and seems to depend for its recognition on the analyst's intuitions in particular circumstances.

In summary, both story grammars and discourse analysis provide intuitive insights with regards to the techniques used to *assemble* a text's subject matter. These considerations, however, are not sufficient in order to explain what a text is about, or how it is comprehended.

2.2.2 Text Linguistics

Story grammars constitute an early attempt at characterizing macrostructures in terms of cognitive schemata. These schemata are psychological constructs that orient the reader to the text and guide his interpretation of it.[2] In psychology, several researchers (*e.g.,* Rumelhart, 1975; Kintsch and van Dijk, 1978; Haberlandt, 1980; Kintsch, 1980; van Dijk and Kintsch, 1983; Britton and Black, 1985; Graesser and Clark, 1985) have proposed a multitude of different schemata and corresponding comprehenders for textual analysis. These models assume that the text-understanding process is controlled by a macrostructure, and rely on complex, purely semantic, mechanisms (*e.g.,* coherence graphs, supervised application of macrorules, etc.) lacking computational principles.

Research on text comprehension in psychology and NLP has been combined under the umbrella term *text linguistics* and generally adopts *schema-based* models. Christopher Habel (1983) points out:

> Most of the work done ... involving the investigation of larger textual units, *i.e.*, beyond the level of sentences, has to do with 'stories'. But these 'stories' are different from those texts which literary critics study.... Most of the stories are not authentic, *i.e.,* they were produced by the researchers themselves in order to test the system in question. [Also,]

[2] Kintsch and van Dijk (1978, p.373) remark that "if a reader's goals are vague, and the text that he or she reads lacks a conventional structure, different schemata might be set up by different readers, essentially in an unpredictable manner.... In many cases, of course, people read loosely structured texts with no clear goals in mind. The outcome of such comprehension processes, as far as the resulting macrostructure is concerned, is indeterminate."

> they are restricted with respect to several important parameters, among others: the length of the text, the vocabulary used, and the domain of the stories.

The difficulty with schema-based models that use macrostructures to account for a text's structure and subject matter is that *ad hoc* conceptual constructs (the macrostructures) and algorithms are postulated to fit the text and then used to explain it. In other words, the description of macrostructures is conflated with an explanation of the comprehension process.

For example, let us consider the distinction (Garnham, 1983; Vipond and Hunt, 1984) between a *story* and a *narrative*. In his work, Wilensky (1982, 1983a, 1983b) introduced the notion of *story points* to try to enforce this dichotomy. Points are schemata that specify those things that a story can be about. They characterize the content that can constitute 'reasonable' stories and account for the existence of a story as an item to be communicated. A person tells, or listens to, a story because it has a content of some intrinsic interest. The content that bears this interest value is termed the *point*. A text that does not possess story points is not considered to be a story. Two kinds of points can be distinguished. An *external point* is some goal that a storyteller may have in telling a story (*e.g.*, to entertain). An *internal* or *content point* is some part of the story itself that generates interest. Wilensky's model is limited to content points. Other researchers have focused on external points (*e.g.*, Schank, Collins, Davis, Johnson, Lytinen and Reiser, 1982). Story points constitute the macrostructures of Wilensky's model of comprehension: they define relevance and, therefore, they implicitly specify what a text may be about (*i.e.*, its subject matter).

BORIS, the program written by Michael Dyer (1983), relies on no less than seventeen different types of conceptual structure. In particular, the same way story points are the macrostructures of Wilensky, Dyer's thematic abstraction units (TAUs) are predefined macrostructures that enumerate the possible gists of a narrative. Terry Winograd and Carlos Flores (1986, p.122) comment:

> If we examine the workings of BORIS we find a menagerie of script-like representations that were used in preparing the system for the one specific story it could answer questions about.

All schema-based approaches to text comprehension share common technical problems (Norvig, 1983a, 1983b; Birnbaum, 1985). Some researchers (*e.g.*, Thorndike and Yekovich, 1980; Alba and Hasher, 1983) also criticize the corresponding theories of human linguistic memory. Text linguistic models of understanding all view a text as consisting of a coherent sequence of sentences. *Coherence* is a *semantic* relation, that is, it applies to meaning; *cohesion* is a *structural* one (*i.e.,* it applies to the symbols). From this viewpoint, schema-based models are primarily semantic approaches to language and meaning.[3] Each sentence is considered as expressing one or more ideas. Coherence of the text means that these ideas are related one to another and 'make sense' together. Often, these relations are conditional: one idea will make another possible, probable, or necessary. *Local coherence* allows the reader to perceive parts of the sequence of sentences as a set of related ideas; *global coherence* emphasizes the notion of a text as a whole, and thus is directly related to subject matter. It is generally agreed that to preserve the local impression of coherence, the reader must constantly use prior knowledge to infer the implicit information that is necessary to 'bridge' from one sentence to another. Arthur Graesser and Leslie Clark, who use the term *bridging inferences* (1985, pp.28–30), remark that there is a consensus regarding the existence of such inferences.[4] Let me illustrate this notion of bridging inference using one of Wilensky's (1983b) examples:

Example 2.2.1 *Willa was hungry. She picked up the Michelin guide.*

Here is a possible sequence of bridging inferences assumed to be necessary to understand these sentences:

- The word 'hungry' represents an instance of the *concept* HUNGER.
- HUNGER creates the *goal* of EATING.
- The EATING goal must be satisfied by the EATING *plan.*
- The EATING plan requires knowing the location of FOOD.
- FOOD is available at a RESTAURANT.
- EATING at a RESTAURANT requires finding one.

[3] With the exception of Steven Lytinen's (1984) work, schema-based text linguistics models have generally ignored the role of syntax during comprehension (*e.g.,* van Dijk and Kintsch, 1983; Dyer, 1983).

[4] See Graesser and Clark (1985, pp.17–32), Rickheit, Schnotz, and Strohner (1985), and Norvig (1987, chapter 2) for a review of the different types and taxonomies of inferences proposed in psychology and artificial intelligence for text understanding. A survey of these references will demonstrate the multitude of different models proposed even for a specific type of bridging inference (*e.g.,* temporal and causal connectives).

- The MICHELIN GUIDE satisfies the goal of finding a RESTAURANT's location.

In fact, this sequence could be even more fine-grained and by no means are bridging inferences limited to this type of inference. The point is that bridging inferences are the rules of composition of schema-based approaches: the meaning of a word is determined by the set of (syntactic and semantic) schemata it refers to, and the meaning of a larger linguistic element is determined by the rules of composition (*i.e.,* the bridging inferences) used to combine the schemata referred to by the words that form this linguistic element.

With respect to bridging inferences, existing computational schema-based models of text comprehension can be partitioned into two principal categories: the first uses models in which bridging inferences are predefined in schemata (*e.g.,* Schank, 1972, 1982; Cullingford, 1978; Lebowitz, 1980, 1988; DeJong, 1982; Dyer, 1983). In this case, understanding is reduced to matching the input against an *a priori* set of schemata, but few schema-based models have schemata that account for the global coherence and gist of a text. Generally, schema-based models can process only texts that closely match their schemata, and they fail badly unless there is a close match. Recognizing these flaws, Alex Kass (1986) and David Leake and Christopher Owens (1986) propose a system, SWALE, that 'learns' schemata by modifying old ones in order to understand 'anomalous' events in stories. Like all programs in the Schankian tradition (see Schank, 1982), SWALE heavily relies on the notion of 'anomaly' (*e.g.,* goal failure) and failure-driven memory. In fact, SWALE consists of a schema-matching model in which some schemata are 'meta-schemata' used to *tweak* (*i.e.,* modify) simpler ones. (The tweaking schemata cannot modify other tweaking schemata). Leake (1989) uses the notion of anomaly to develop ACCEPTER, yet another schema-based story understanding program, which uses gradual anomaly detection strategies.

The other group of schema-based models builds and evaluates inference chains at reading time (*e.g.,* Rieger, 1975; Charniak, 1983; Granger, Holbrook, and Eiselt, 1983; Wilensky, 1983b; Alterman, 1985; Riesbeck and Martin, 1985; Norvig, 1987, Martin, 1989). *Inference chaining* is based on *marker-passing* architectures (see section 1.1 and, Anderson, 1983; Hendler, 1987, 1989), that is, models of associative semantic memory in which the information (*i.e.,* the markers) exchanged between the elements of memory is either very simple as is the case of Fahlman's NETL (1979), or complex (possibly including control information) (*e.g.,* Charniak, 1983, 1986a, 1986b; Hirst, 1987). Some inference-chaining models, for example Norvig's FAUSTUS (1987), proceed directly from Wilensky's PAM, in that they specify the knowledge required for the evaluation of paths in schemata. Others (*e.g.*, Stallard, 1987; Charniak and Goldman, 1988; Hobbs, Stickel, Martin, and Edwards, 1988; Pollack and Pereira, 1988) postulate a *path checker* module that provides a formal truth-based algorithm for evaluating inference chains.

The basic strategy for inference chaining consists of generating chains (or paths) of bridging inferences at reading time and somehow evaluating these paths to decide which provide a 'good' explanation. For example, Wilensky's PAM program attempts "to match inputs to known goals [and plans] of the actor, and to backward-chain to these goals if they [can't] be matched directly" (Kass, 1986).

Inference-chaining models are typically limited to local coherence and do not address the problems of global coherence or of subject matter. Also, recall that these models can be intractable or may generate an unmanageable number of useless inferences while trying to obtain 'relevant' (or 'important') inference paths (Norvig, 1989).

2.2.3 Lexical Statistics

The basic problem with the methodologies reviewed above is that either they are restricted to the local scale, or they arbitrarily specify a set of *a priori* gists. Researchers interested in computer-assisted literary analysis resort instead to a 'knowledge-free', statistical approach to subject matter. The fundamental postulate of *lexical statistics* is that the choice of vocabulary in a text is largely a function of subject matter. More precisely, these researchers assume that the distribution of lexical patterns over large amounts of text directly correlates to subject matter (*e.g.*, Muller, 1977; Phillips, 1985; Ide, 1986).

In the simplest form of lexical statistics, the patterns are individual words and the statistics are limited to frequency counts. This technique is problematic. For example, a certain word may be repeated *ad nauseam* in one chapter of a book without having any major impact on the perception of the subject matter of the whole text. In other words, the problem inherent to any straightforward frequency-count approach is that there is not sufficient justification to correlate frequency of occurrence to subject matter. Conversely, a strictly distributional approach, that is, one that measures the distribution of a pattern over a text, is also inadequate, as a certain lexical pattern may appear throughout a text and yet not often enough to affect aboutness. Thus, researchers in lexical statistics typically use a combination of both frequency of occurrence with distribution. This methodology is taken to be more meaningfully reflective of the subject matter of a text than the consideration of either characteristic alone.

In order to *interpret* the results of such statistical analyses, researchers require the absolute *probability of use* of each pattern. For example, a word with low absolute probability of use need not appear as often as one with high probability to be marked as relevant. The notion of probability of use, however, is problematic. Consider Muller's (1977, p.46) warning:

> The notion of probability of use should be applied only to a lexis of situation, and not to the lexis of the individual, still less to the lexis of the collectivity.

In other words, "the notion of an individual word, without regard for its context, as the identifiable signifier for a particular concept or meaning is problematic" (Ide, 1986). (This fundamental observation is not restricted to lexical statistics but, as we shall see in the next section, concerns the whole information-processing paradigm.) Since lexical statistics operates at the lexical level of analysis, the notion of *situation* (or equivalently, *context*) must be defined only in terms of lexical items. Consequently, the concepts of *node*, *span*, and *collocation* are introduced (Phillips, 1985, pp.43–44):

> The *node* refers to the lexical item in the focus of attention, *span* relates to the number of items in the immediate [linear] context of the node and *collocation* is the term used to denote a common co-occurrence of items within a given span, that is, the joint occurrence of a node and a particular *collocate*.

With regards to the notion of span, Phillips (*ibid.*) remarks that "it is crucial to have a clear idea of how far the 'influence' of a word extends into its syntagmatic environment since this determines the limits within which patterns of association are to be sought. It was found that a span setting of four orthographic words on either side of the node yields optimum results."

The immediate difficulty with lexical statistics is that the determination of its analytical categories (*i.e.*, the nature and length of the lexical patterns to be scrutinized) is typically carried out by hand. In other words, the task of identifying patterns that refer to a common concept or theme ultimately depends on the intuitions of an expert. The interpretation of the statistical results also rests on an expert. And, most importantly, lexical statistics presents the problem of ignoring subtle (semantic) relations among the words of a text that may significantly affect the perception of subject matter. For example, neither pronoun comprehension (Hirst, 1981; Stevenson, 1986) nor lexical disambiguation can be tackled without some semantic 'knowledge'.

In summary, in dealing only with the *surface text*, that is, with the stream of characters that constitutes the text, lexical statistics is inherently restricted to analyses of the frequency and distance between configurations of lexical items. The point I want to stress is that lexical statistics consists in an *a posteriori knowledge-free analysis* of a text and thus cannot serve as a cognitive model of text comprehension. For this reason, I shall not discuss this methodology any further.

2.3 On the Existence of Macrostructures

I have already stated that, within the information-processing paradigm, all cognitive approaches to the perception of subject matter assume the existence of certain global patterns of textual organization that have been called *macrostructures* (section 2.2.2). The analytical categories and the postulated macrostructures of these approaches stem from a strictly psychological methodology: the schemata, which capture the rules of a given model, are derived from empirical methods. In other words, statistical analysis directs the design of the rules. This is particularly obvious in Dyer's (1983) and Graesser and Clark's (1985) work. Beyond methodological issues, the problem with this approach is that it is not clear that laboratory experiments do not set up an artificial environment that can affect the results. Rand Spiro (1980), for example, argues that if the experimenter tells the reader what he intends to ask after the reading of a text, then the reader may obtain different results for recall and interpretation. (Also see Mitchell, 1982, pp.102–103.) Spiro's conclusions have important repercussions.

Text linguists assume *a priori* that there is a unique, small, correct set of macrostructures (*e.g.*, Dyer's thematic abstraction units, Lehnert's (1981) plot units). According to them, the perception of subject matter simply consists in finding out which macrostructure(s) the text corresponds to. From this viewpoint, comprehension is *normalized*. Normalization implies an authoritative *expert* who sets the standard, that is, who decides what is normal and what is not, what constitutes a correct interpretation and what does not. In other words, an expert is needed to interpret the results and specify rules from them. Spiro's claim is that the expert can condition the results (that is, how a reader reports his or her comprehension of a text) through the design and control of the experience itself. From this observation, George Dillon (1980) simply rejects the existence of macrostructures. For him, only the expert (as opposed to a 'natural unconditioned' reader) seeks macrostructures (*i.e.,* rules of interpretation) in a text.

The point to be grasped is that researchers generate rules that may not *directly* apply to comprehension, but rather to an *a posteriori* or an artificial *expression* of meaning and subject matter. It is important not to confuse information that is actually stored in memory with reconstruction of information on the basis of inferential reasoning (Wagenaar, 1988). Also, the difficulty with the notion of macrostructure stems from the absence of a validation method to guide the specification of rules encoded in a model. This situation is reflected in the following comment from Graesser and Clark (1985, p.1):

> There is widespread disagreement about *what* inferences are generated, *when* inferences are generated, *how many* inferences are generated, and *what knowledge sources* contribute to the generation of inferences.

Elke van der Meer (1987, figure 6, p.51) makes another important remark in arguing that the types of bridging inferences used during comprehension (especially inferences about time as it pertains to the story line) are *not* the ones that researchers in text linguistics have been studying (*e.g.,* causality, consequence, finality, and superordination).

In summary, the existence of a small, correct set of macrostructures on which existing information-processing models of text comprehension depend is problematic.

2.4 General Objections to the Conduit Metaphor

2.4.1 Beyond Literal Meaning: Tractability and Context

I previously stated that, in the framework of the conduit metaphor, a common view holds that a sentence has a literal meaning. Some researchers reject the assumption that obtaining the literal meaning of an utterance is a necessary step on the path to understanding. Raymond Gibbs (1984), for example, writes that "the literal meaning of a sentence is an inadequate place to start figuring out an utterance's meaning". Searle (1979) argues that literal interpretation can only account for the meaning of some sentences. Consider, for example:

Example 2.4.1 *John quickly cut through the red tape.*

John may literally be cutting a red tape in order to unwrap his Christmas gift, or John may be particularly efficient in his dealings with bureaucracy.

Graesser and Clark (1985, p.27) report that:

> There has been some debate in psychology about the time course of interpreting the literal meaning of a request versus the intended (illocutionary) meaning of a request.... According to one alternative, the comprehender first interprets the literal meaning and subsequently infers the illocutionary meaning by integrating the literal meaning with the context of the speech act. A second alternative is that the process of constructing a literal meaning and the process of constructing an illocutionary meaning are executed simultaneously. A third alternative is that the illocutionary meaning is directly interpreted and that the literal meaning may not be interpreted in some contexts.

From the above example, it seems that the issue is not limited to requests but, in fact, extends to any utterance whose meaning *may* depend on context (see Dascal, 1989; Gibbs, 1989).

A most fundamental problem hides behind the immediate methodological puzzle: meaning comes from rules and it seems that some rules *must* consider context since the *signifiant* and the literal meaning are not enough in certain cases to obtain the *signifié*. It follows that the 'single-sentence' paradigm is oversimplistic. Consider the following example:

Example 2.4.2 *It is certainly getting hot in here.*

This sentence has at least four interpretations![5] It can be understood literally, but this provides no clue as to which interpretation is adequate. The question then is to know where meaning comes from. The conduit metaphor explicitly claims that the meaning is *in* the words, but apparently this is not the case: words are not enough, we need context. This remark is not limited to the meaning of sentences, but also applies to the meaning of individual words. For example, consider the word *red* in the following idioms:

- Red carpet, red tape, red light, red light district, red meat, red wine, redhead, red herring.
- To see red.

[5] For example,

1. **Literal**: It's hot in here.
2. **Speech act**: It's too hot in here, could you open a window.
3. **Ironical speech act**: It's freezing in here could you close the door.
4. **Figurative**: This discussion is degenerating into a bitter argument.

- To be in the red.

- To catch someone red-handed.

In these examples, the meaning of the idiom does not proceed from the individual meanings of its elements; the idiom must be treated as an indivisible semantic whole. The problem then is to recognize an idiomatic meaning from a literal one. For example, as explained above, *red tape* can be taken literally or idiomatically; only context may allow one to discriminate between the two usages.

I want to emphasize that contextual influence on meaning is not restricted to a small set of idioms, but on the contrary permeates language. Consider, for example, the utterance:

Example 2.4.3 *Wylbur is a pig.*

A dictionary (*e.g.*, Webster, 1981, p.862) suggests *some* possible interpretations; for example:

- Wylbur is a young swine not yet sexually mature.

- Wylbur is an immoral woman.

- Wylbur is a policeman.

But a dictionary does not exhaustively list all possible uses of the word *pig* and, therefore, typically provides an extremely vague definition such as "one resembling a pig"[6] to account for these idiosyncratic uses. Consider, for example, other possible interpretations of the above utterance:

- Wylbur eats like a pig.

- Wylbur is some sort of sexual maniac.

- Wylbur has poor bathing habits.

- Wylbur simply did something I resent.

The contextual view of meaning is associated, in British linguistics, with the name of J.R. Firth. For him, language is essentially a social and conventional phenomenon. Text

[6] An analogical definition is problematic because the perception of the *soundness* or *felicity* of an analogy largely depends on the comprehender.

is viewed as the only immediate component of a context of situation that lends itself to analysis. Context of situation, which Firth regards as the prime analytical category, is an abstraction of a system of relations from the life of humans in society. Firth (1957) argues that the complete meaning of a word is *always* contextual. Phillips (1985, p.14) observes that this leads Firth to the *apparently* extreme position of considering each use of a word in a new context as an occurrence of a new word. Similarly, Firth (*ibid.*) remarks:

> An isolated word which does not function in a context of experience has little that can be called meaning.

Contextual meaning is highly problematic within the conduit metaphor in which typically meaning is decontextualized and comprehension is reduced to an invariant algorithm that ignores all *subjective* (or equivalently, *private* or *idiosyncratic*) aspects of the act of interpretation itself. Let me justify this observation. Within the information-processing paradigm, it is generally assumed that the meaning of a sentence can be 'reasoned out', that is, obtained by means of rules of inference (that can be content-blind, as in the case of *modus ponens*, or content-dependent as in "if you are hungry then you need food"). These rules of inference operate on symbols that are taken *a priori* to be context-dependent (or equivalently, domain-dependent). The difficulty with such an approach is that the postulated rules only have an *a posteriori* explicative nature, that is, they do not address the problems, first, of deciding which symbols (words and sentences) *may* have a non-literal meaning, and second, of *discriminating* between the several possible contextual meanings of a symbol. In other words, the context (or domain of discourse) is a given and the issue of recognizing it is altogether bypassed. Some researchers (*e.g.*, Schank and Abelson, 1977; Dyer, 1983) tackle the problem of contextual meaning by advocating the use of a *scriptal lexicon*, which tries to specify rules of recognition for all possible contexts. This approach is still problematic, as Lawrence Birnbaum (1985) remarks:

> No single explanatory inference rule can be expected to attend to all the aspects of a situation which might affect the truth or relevance of the explanation it offers.

In other words, it is quite frequently (if not always) possible to present an example that violates a context-recognition rule. Therefore, within the conduit metaphor, the problem of contextual meaning is typically transferred to the individual words of a sentence.

The first difficulty with doing that comes from the lack of availability of an accepted *exhaustive* list of meanings for words. Dictionaries constitute one possible source for the definition of words. But Firth (1957) argues vigorously against the view of meaning as somehow 'contained' in words that 'express' the meanings enshrined against their written forms in dictionaries. Assuming that words have lexical meanings that anyone can access by consulting a dictionary offers only a very partial solution to the problem of meaning, for people seldom use words in such a rigid way, as the preceding examples demonstrate. Moreover, the organization of a dictionary and, consequently, of the lexicon of a model, is problematic (Miller, 1985). Ultimately, we require an expert to *standardize* meaning, that is, to specify an *a priori* lexicon. This is a formidable, if not impossible task. 'Abstract' words such as *love* and *freedom* generally have vague definitions. Indeed, the definition of *language* that is quoted at the beginning of this chapter uses the term *idea*, whose meaning has been debated by philosophers for centuries (Adler, 1985, chapters 1–3)! (The meanings of the words *meaning* and *understanding* are themselves problematic!) In other words, the *explicit*, *precise* definition of certain words seems very difficult—and impossible when we acknowledge that, despite the existence of dictionaries, the *actual uses* of a multitude of 'concrete' words seem to escape characterization. Consider, for example, the verb *fly* in the following hypothetical dialogue (Stephen Regoczei and Edwin Plantinga, personal communication):

- How can you tell a bird from another animal? *Birds fly.*
- Do birds fly all the time? *No.*
- Do little chicks fly? *No.*
- Do dogs fly? *No.*
- Do dogs on airplanes fly? *Well, yes. . . .*
- Do I fly? *No.*
- Do I fly when I am flying to Montreal. *Well, yes...*
- Do flying squirrels fly? *Yes.*
- Do bats and insects fly? *Yes.*

Does the above dialogue merely play on two *clear, distinct* meanings of the word *fly*? Etymological considerations suggest that, typically, an *often lost* and mostly analogical shared element of meaning underlies most uses of a word. If this was not the case, a rich vocabulary would generally provide more-precise words to distinguish these uses. In other words, it is not clear whether the example above illustrates two different meanings or two *shades* of a same meaning.

In the same vein, John Sowa (1984, p.346) observes that "even the boundary between [a] tree and [its] environment may be indistinct: the tree may have started as a sprout from the root of another tree and may still share a root system with its parent and siblings." Does a tree growing from another one constitute a different tree? Generally, the semantic distinctions we make are not adequate in 'boundary cases'. The notion of *death* in its common, medical, and legal uses exemplifies this remark.

The need to avoid an *a priori static* classification of the uses of words is echoed in the following four principles of language taken from Jack Odell's (1984) list of twelve:

- *Open Texture*: Even if we legislate sets of necessary and sufficient conditions to govern what [words] mean, we can't be sure that our legislations preclude the existence of contexts where we will be uncertain what our words mean, that is, we can still *imagine* cases where we wouldn't know whether or not a given word applied.
- *Creativity*: We use language in inventive and innovative ways to amuse, clarify, convince, annoy, insult, etc. Punning, poetry, word play, and pre-eminent prose all depend on our ability to use language with a certain impunity.
- *Family Resemblance*: [(Wittgenstein, 1953)]: What most, if not all, general empirical terms *mean* in natural language, as opposed to what we might *mean by* them on specific occasion, cannot be specified formally, that is, in terms of necessary and sufficient conditions.
- *Non-Functionality*: What a given string of words means is not a function of the formal characteristics that string possesses. "Why not?" can be used to make a request, even though its *form* is that of a question.

In summary, it is not clear that an exhaustive list of all possible contextual uses of an extended set of linguistic elements may be achievable.

The second problem with having to resort to such an exhaustive lexicon is that it leads to an inescapable trade-off between representational blindness and algorithmic intractability, which I will now briefly explain. On the one hand, all possible (literal, idiomatic, and figurative) meanings of a word must be stored in some explicit representations (either in the schemata of the lexicon or the rules of the understanding algorithm). Even if we restrict ourselves to the definitions given in a dictionary, each word will 'point' (or refer) to a large number of meanings. On the other hand, if a word does not point to a small number of meanings, the model will be faced with an intractable number of inferences (*i.e.*, compositions of meanings) at understanding time.

In a computational framework, intractability is totally unacceptable. It follows that the implemented models of linguistic comprehension restrict the number of meanings of each word in one way or another. Schema-matching models simply use small lexicons. Inference-chaining models typically postulate the sort of context-blind interpretation rules (*e.g.*, Wilensky's (1983b) *meta-plans*) that I have criticized above. The difficulty with such solutions to the problem of contextual meaning is that they lead to an *artificial* (*i.e.,* fixed-patterned) language outside of which 'comprehension' is impossible. Though in computer science it is usually the case that the human user must resort to an artificial language to communicate with the machine, this is not acceptable in the case of *natural* language processing. Winograd and Flores discuss this issue at length and conclude that models that legislate contextual meaning merely create "a narrowed microworld that reflects the blindness of [their] representation[s]" (1986, p.123).

The trade-off between intractability (resulting from an exhaustive lexicon) and blindness (of the representations used in models that considerably limit contextual meaning) is best exemplified by a most counterintuitive feature of all schema-based models: the more these models 'know', the slower they become. This is in direct contradiction with the commonly accepted assumption that the more familiar an input is, the quicker it should be processed.

Recapitulating, I remark that the consideration of the role of context appears to lead to a tremendous increase in the number of rules or schemata that an information-processing model must specify. In turn, the resulting increase in processing complexity leads to a choice between inherent intractability and representational blindness (*i.e.,* oversimplicity).

2.4.2 On the Existence of Rules: The Connectionist Attack

All theories of language and meaning that proceed from the conduit metaphor postulate that linguistic comprehension involves *rules of composition* (*e.g.,* grammars, schemata, inference engines) over symbols. From this point of view, these theories consist of *a priori* symbolic manipulations. Recently, the existence of such *rules of language* and *rules of thought* has been challenged by the *connectionist* paradigm (Rumelhart, 1984, McClelland and Rumelhart, 1986).

The fundamental hypothesis of the connectionist approach (Feldman, 1984, 1985a, 1985b) is, as mentioned in section 1.1, that if we acknowledge that the human brain is involved in the act of understanding, then the consideration of its anatomy and physiology may provide helpful insights for the design of a computational model of cognition.

Feldman (1984) summarizes the connectionist attack on the information-processing paradigm:

> One consequence of taking [biological] computational constraints seriously is a profound reservation on the ultimate viability of many of the information-processing models cur-

> rently dominating the field. Any paradigm that depends on central control, data structures or symbol manipulation presents the problem of having no obvious reduction to the underlying computational system. Researchers motivated by biological constraints have tended to work on positive results rather than argue paradigms and have been exploiting insights gained through traditional approaches. But it does seem likely that many problems that appear intractable in conventional information-processing paradigms will be accessible in a more natural formalism[.]

Indeed, some connectionist models appear to handle contextual meaning in a most natural way, which contrasts with the trade-off inherent to schema-based approaches.

The small number of computational steps assumed for hard recognition problems (such as letterform recognition) has led connectionist researchers to postulate the existence of large, pre-connected networks of simple computing elements operating in *parallel.* It is generally accepted that the human brain also works in a highly parallel fashion. Conversely, the von Neumann computer used by most existing computational models of NLP is sequential. But the issue at hand is not whether parallelism is essential for understanding but, as Zenon Pylyshyn (1984a, 1984b) remarks, whether or not cognition should be *characterized* in terms of a formal computation, that is, in terms of symbols and rules.

The on-going debate between information-processing and connectionism is not directly relevant to this dissertation and I will discuss it only briefly. First, it has been observed that some connectionist models are *rule-based* symbolic systems in disguise (Derthick and Plaut, 1986). Second, connectionist researchers use highly idealized models of neurons. Third, several technical problems (*e.g.,* variable binding, frame selection) have been identified with connectionist architectures (Barnden, 1983; Birnbaum, 1985; Pinker and Prince, 1988). In particular, Fodor and Pylyshyn (1988) argue that connectionist representations are unstructured, atomic, and bounded. These claims have been rejected (*e.g.,* Elman, 1989) and solutions for problems such as variable-binding and multi-place predicates have been recently proposed (*e.g.,* Ajjanagadde and Shastri, 1989; Anandan, Letovsky and Mjolsness, 1989). Fourth, as mentioned in section 1.1, connectionist models for linguistic comprehension (*e.g.,* Cottrell, 1984, 1985; Selman, 1985; Waltz and Pollack; 1985; Berg, 1987; Selman and Hirst, 1987; McClelland and Kawamoto, 1986) are typically single-sentence parsers that produce a pattern of activation corresponding to a parse tree, and few can tackle simple lexical and structural disambiguation, or even very simple inferences (*e.g.,* Eiselt and Granger, 1987). Finally, there have been recent attempts to integrate connectionism with symbolic marker-passing (*e.g.,* Chun and Mimo, 1987; Hendler, 1989; Lange and Dyer, 1989; Lange, Hodges, Fuenmayor, Belyaev, 1989; Lee, Flowers, and Dyer, 1989), but these models are, in fact, information-processing models that are not directly concerned with the computational constraints of neuronal modeling, and they still generally mostly rely on complex symbolic data or processes.

The debate between connectionism and information-processing is a specific instance of a more general and philosophical question, the *mind-body* problem (Johnson, 1987), which Paul Thagard (1986) describes:

> The currently dominant position in the philosophy of mind is *functionalism*, which says that mental states are to be understood in terms of their functional relationships to other mental states, not in terms of any material instantiation.... The rejection of a direct mind-matter link distinguishes functionalism from the *mind-body identity theory*, according to which types of mental states such as thoughts are identical to types of states in the brain.... In computational terms, functionalism is the claim that only software matters to the mental. The argument for multiple instantiation [on which functionalism is based] says that we can ignore hardware in characterizing the mental, since the same software can run on any number of different kinds of hardware: It is the functional performance of the software which is crucial.

The information-processing paradigm corresponds to a functionalist approach, whereas the mind-body identity theory underlies connectionism.

Functionalism has not been without its critics, even in philosophy. For example, Paul Churchland (1985) advocates *eliminative materialism*, which claims that advances in neuroscience will lead us to a very different set of categories for describing mental states, eliminating the old ones. Is there an obvious, verifiable solution to the mind-body problem? Thagard (1986) writes:

> My conclusion is that we currently know too little about the human mind and brain and about the range of possibility of other kinds of intelligence to form a plausible solution to the intelligence-matter [or equivalently, mind-body] problem. Any answer offered at this point would be a generalization from one ill-understood instance, the brain.

In other words, our current understanding of the brain allows us only to *assume* a certain organization for the mind and the brain of a reader; experimental psychology, genetics, and neurology are a long way from verifiable and unanimously accepted cognitive theories. Indeed, each of the existing theories of comprehension has a *basic metaphor* underlying it and Earl Mac Cormac (1985, p.17) warns us that "metaphors can be dangerous not only in bewitching us into thinking that what they suggest really does exist but also in leading us to believe that the attributes normally possessed by any of the referents in the metaphor are possessed by the others."

The computational metaphor (*ibid.*, pp.9–22) underlies most, if not all, existing research in linguistic comprehension. Under this metaphor, the brain is viewed as a computational device similar to a computer, and the mind emerges as a series of *algorithms* by means of which the brain functions. Algorithms are the essence of the 'word expert' approach adopted by Steven Small (1980, 1983; Small and Rieger, 1982) for sentence parsing: each word in the lexicon is represented by a procedure and the parsing and semantic interpretation of a sentence are performed through the interactions of the concurrently running procedures. A general mechanism of comprehension is abandoned in favor of a very large number of loosely related experts (Hirst, 1987, section 4.2.4).

I investigate the repercussions of the computational metaphor on a theory of interpretation in the next section.

2.5 Beyond Algorithmic Competence

2.5.1 Text Linguistics as a Structuralist Theory

The conduit metaphor views linguistic expressions as containers for meanings. The role of the reader is to retrieve the *determinate* meaning placed in a text by the writer. To do so, it is postulated that the reader must bridge from one sentence to the other by generating inferences: "the crucial problem of story understanding is inference" (Kass, 1986). Most of the existing computational NLP models consider only these bridging inferences. In the context of *text* comprehension this assumption presents the disadvantage of being too compartmentalized, that is, of ignoring the problem of the perception of subject matter: the bridging inferences presented in those models are restricted to the level of local coherence. This is not enough, as Dyer (1983, Preface) remarks:

> In-depth understanding means being able to do more than simply extract the facts of a narrative and infer causal connections between them. An in-depth understander must be able to recognize what was memorable about a narrative, what episodes were of significance, and what the point of the narrative was—that is, why the narrative was worth telling in the first place. Finally, if a narrative is significant in some way, the memory must be updated so that it will come to mind in appropriate future situations.

I previously observed that current information-processing theories that consider the problem of subject matter postulate a fixed set of macrostructures that, in essence, specify the possible gists of a text. The use of macrostructures constitutes a *structuralist* approach to comprehension. Let me first quote at length Christopher Norris's (1982, p.2) summary of what structuralism is generally taken to be:

> The concept of *structure* serves to immobilize the play of meaning in a text and reduce it to a manageable compass. This process can be seen at work in the reception of a book like Jonathan Culler's *Structuralist Poetics* (1975), regarded ...as a sound and authoritative guide to the complexities of structuralist thought.... The proper task of theory, in [Culler's] view, is to provide a legitimating framework or system for insights which a 'competent' reader should be able to arrive at and check against his sense of relevance and fitness.... His argument becomes strained when it tries to link this notion of readerly 'competence' with an account of the manifold conventions—or arbitrary codes—that make up a literate response. On the one hand, Culler appeals to what seems a loose extension of ...Chomsky's argument: that linguistic structures are innately programmed in the human mind and operate both as a constraint upon language and as a means of shared understanding. Thus Culler puts the case that our comprehension of literary texts is conditioned by a similar 'grammar' of response which enables us to pick out the relevant structures of meaning from an otherwise inchoate mass of competing detail. On the other hand, he is obliged to recognize that literary texts ...involve certain specialized codes of understanding which have to be acquired.

Existing information-processing theories of comprehension correspond to a structuralist methodology in that they try to produce a set of rules that would encapsulate this innate readerly *competence*. These theories are also *objectivist* (Winograd and Flores, 1986, p.28):

> For the objectivist school of hermeneutics, the text must have a meaning that exists independently of the act of interpretation. The goal of a hermeneutic theory (a theory of interpretation) is to develop methods by which we rid ourselves of all prejudices and produce an objective analysis of what is really there. The ideal is to completely 'decontextualize' the text.

I have already mentioned Dillon's (1980) rejection of the notion of macrostructure *per se*. Let us investigate, in the next subsection, two fundamental problems that pertain to the notion of 'innate readerly competence'.

2.5.2 On Readerly Competence

On Innatism

According to Culler's (1975) most conservative view of structuralism, the reader possesses an innate *competence* that allows him to perceive what is relevant and what is not. From this point of view, Culler is indeed very close to Chomsky's position on language and cognition (in Piatelli-Palmarini, 1980, p.10):

> The environment per se has no structure, or at least none that is *directly* assimilable by the organism. All laws of order, whether they are biological, cognitive or linguistic,

> come from inside, and order is *imposed* upon the perceptual world, not *derived* from it. These laws of order are assumed to be species-specific, invariant over time and across individuals and cultures.

Do the apparent regularities in linguistic comprehension originate in innate rules or are they derived from the environment (and, in particular, from a culture and one or more linguistic communities)? A biological discussion of the merits of the innatist hypothesis lies beyond the scope of this dissertation. Let me simply remark that the debate between innatists (or equivalently, nativists) and researchers who postulate some assimilation of ideas from the environment started more than 2500 years ago (when Aristotle rejected the *forms*, or universal archetypes, of Plato and advocated a *tabula rasa*) and still rages (see Piatelli-Palmarini, 1980; Lakoff, 1987). Jean Piaget (1970) concisely summarizes the issue when he remarks that the boundary between the *phenotype* and the *genotype*, that is, between the acquired (from the interaction of genetics with the environment) and the innate, is *floue* (fuzzy). In the end, the biological debate reduces roughly to an interpretation of the role of *random mutations*, that is, to a debate on the paradoxical notion of probability "which has puzzled philosophers ever since Pascal initiated that branch of mathematics—and which von Neumann, the greatest mathematician of our century, called 'black magic'" (Koestler, 1978, p.266).

The above discussion suggests that the innatist hypothesis merely constitutes yet another basic metaphor. Let me repeat Thagard's (1986) warning: "any answer [to the mind-body problem] offered at this point would be a generalization from one ill-understood instance, the brain".

Structuralism and Skepticism

Immanuel Kant claimed that man must possess certain innate faculties of mind by virtue of which he imposes law and order on his experiences. The laws man discovers in nature are those he puts there himself. It is almost as if man created nature, subject to one important proviso. Kant believed that underlying man's experiences are unknowable *things-in-themselves*, which would continue to exist even if there were no minds left. The role of the mind is to organize the things-in-themselves into forms that make experiences intelligible. The result is man's perception of nature. Kant regarded the active but unconscious mental powers of man to be *a priori*, that is, to exist in the mind prior to experience, although not as ideas since, he claimed, the content of ideas can come only from sensory experience. Norris (1982, pp.4–5) remarks:

> It is not hard to see the parallels between Kantian thought and the structuralist outlook presented by a theorist like Culler. Both have their origins in a sceptical divorce between mind and the 'reality' it seeks to understand. In structuralist terms this divorce was most clearly spelled out by the linguist Ferdinand de Saussure. He argued that our knowledge of the world is inextricably shaped and conditioned by the language that serves to represent it. Saussure's insistence on the 'arbitrary' nature of the sign led to his undoing of the natural link that common sense assumes to exist between word and thing.... In his view, our knowledge of things is insensibly structured by the systems of code and convention which alone enable us to classify and organize the chaotic flow of experience. This basic *relativity* of thought and meaning ...is the starting-point of structuralist theory.

The theme of the relativity of language and meaning is typified in what has come to be known as the 'Sapir-Whorf' hypothesis. Benjamin Whorf (1956, p.213) writes:

> We dissect nature along lines laid down by our native languages. The categories and types we isolate from the world of phenomena we do not find because they stare every observer in the face; on the contrary, the world is presented in a kaleidoscopic flux of impressions which has to be organized by our minds—and this means largely by the linguistic systems in our minds. We cut nature up, organize it into concepts, and ascribe significances as we do, largely because we are parties to an agreement to organize it in this way—an agreement that holds throughout our speech community and is codified in the patterns of our language.

The Sapir-Whorf hypothesis (*e.g.,* Anderson, 1980, pp.384–386) claims that the language of an individual *partially* determines the world view and the conceptual system of this individual. Whorf became convinced of this hypothesis after studying Hopi Indians who apparently had no implicit or explicit concept of time. In the same vein, Sowa (1984, p.347) observes that English speakers can easily talk about hypothetical situations that have not happened, but Chinese has no syntactic form for expressing them and, therefore, the comprehension of conditionals (*e.g.,* in English) is harder for a native Chinese speaker.

Against the Sapir-Whorf hypothesis Brent Berlin and Paul Kay (1969) claim that the color vocabularies of various languages form a fixed pattern. Eleanor Rosch (1974) has extended this notion of 'prototypicality' beyond color to other categories (*e.g.,* facial expression of emotions), arguing that humans categorize according to innate prototypes rather than by analyzing the features of objects and classifying them abstractly (see Mac Cormac, 1985, pp.71–72). From this evidence, most researchers dismiss the hypothesis. John Anderson (1980, p.386), for example, observes:

> The evidence tends not to support the hypothesis that language has any significant effect on the way we think or on the way we perceive the world. It is certainly true that

> language can influence us, . . . but its effect is to communicate ideas, not to determine the kind of ideas we can think about.

Most importantly, rejecting *linguistic* relativity does not solve the problem of *conceptual* (or mental) relativity that results from placing structure in mind. The information-processing and structuralist paradigms exemplify the Kantian response "which strives to keep skepticism at bay by insisting on the *normative* or somehow *self-validating* habits of readerly 'competence' " (Norris, p.5). Let us focus on this fundamental remark, which states that structuralist 'readerly competence' is self-validating.

Ultimately, the rules used in a structuralist model of interpretation are grounded in the *expert* and his ability to *abstract* (or classify) from data, that is, to extract *relevant patterns* that are assumed to exist independently of the expert's act of interpretation. Meaning comes from this *objective* recognition of *distinctive* features and *significant* contrasts. The expert can be viewed as creating a master code for the interpretation of text. Structuralism becomes in effect a natural extension or legitimating theory of what it is to read a text properly. It is hoped that the rules of a complete model will capture the structure of the competent reader, that is, the master code he uses to impose meaning.

Roland Barthes (1977) remarks that the language of the expert, what he terms the *meta-language*, is itself an object of study. Norris (pp.9–10) explains:

> Barthes is well aware of the dangers and delusions implicit in a discourse that claims the last word in explanatory power. The semiologist may seem to exercise 'the objective function of decipherer' in relation to a world which 'conceals or naturalizes' the meanings of its own dominant culture. But his apparent objectivity is made possible only by a habit of thought which willingly forgets or suppresses its own provisional status. . . . The dream of total intelligibility, like 'structure' in its metalinguistic sense, belongs (he implies) to a stage of thinking that is self-blinded by its own conceptual metaphors.

In other words, the rules of interpretation (especially the macrostructures), that define a *readerly competence* can only be 'grounded' in the self-validating basic metaphor(s) of an 'expert'. This inadequate response to the basic relativity of language and thought, which underlies structuralism, leaves the door open for the radical skepticism of the deconstructionist movement (see Norris, 1982).

2.5.3 Beyond Algorithmic Control

The rules of interpretations postulated in the existing NLP models come to form an 'understanding algorithm' that defines what it is to correctly understand a text, and what is understandable and what is not. An algorithm specifies a *formal* computation that itself defines a competence, that is, an implicit normative metric. Von der Malsburg (1985) remarks that:

> It is the essence of an algorithm that all its qualitative aspects are premeditated and tested so that during its execution no ideas are necessary [and] no qualitative questions are left open.... Only quantitative decisions must be met, which can be handled in a mechanical way.... There is a very clear division between the qualitative and the quantitative aspects of an algorithm: The former is invented by the human mind and formulated as rules, the latter refers to the data handled by rules.

The point I want to make is that most existing NLP models treat their rules of interpretation as components of an algorithm (as opposed to treating them as data). In other words, the algorithms of these models constitute mechanical encodings of (typically static) sets of rules more or less arbitrarily established by the programmers. This creates the illusion of understanding (Winograd and Flores, 1986, p.123):

> It must be stressed that we are engaging in a particularly dangerous form of blindness if we see the computer—rather than the people who program it—as doing the understanding.

Some philosophers (*e.g.*, Odell, 1984; Searle, 1984) have argued against artificial intelligence as a whole on the basis of this observation. The difficulty with a structuralist hermeneutical approach that is limited to the normative aspect of Culler's theory is that it is blinded by its own basic metaphor(s): the 'readerly competence', that is, the postulated rules of interpretation cannot be validated. Therefore, an algorithmic encoding of such rules is also unacceptable. In other words, I abandon both the hypothesis that *a priori* macrostructures are necessary for text comprehension and the idea that understanding can be characterized by a normative algorithm.

Von der Malsburg observes:

> This conclusion [the abandonment of algorithmic control] creates a dilemma because surely the brain must follow some functional principle and surely this principle can be put into the form of an 'algorithm'.... The solution to this dilemma lies in the scheme of a *trivial algorithm*. All ...rules, values, concepts, methods, procedures, etc., are treated by the trivial algorithm as data. The algorithm fixes the general form of operations on a fundamental level, and makes sure that organized states instead of chaos arise.

The strategy adopted in this dissertation, that is, the use of a 'trivial' algorithm for a model of text comprehension, represents a fundamental shift in concern with respect to existing research. A trivial algorithm is trivial solely in the sense that it is strictly quantitative, that is, free of heuristics (read 'educated guesses') and rules of interpretation. Consequently, there is no *a priori* absolute readerly competence, no *a priori* macrostructures that are specified once and for all in an algorithm: comprehension depends on the data supplied to the trivial algorithm. Within the framework of my working hypotheses, the trivial algorithm corresponds to the *modus operandi* of memory, and, I insist, it is this *modus operandi* that is the object of study of my work. And, although the previously discussed problems of information-processing models are not the issue here, some can be addressed within the scope of a quantitative model of memory for reader-based comprehension:

- The 'blindness' of the user-specified data is inescapable, but can be viewed as partly accounting for the idiosyncratic knowledge of the user.

- The use of parallel networks for memory can help in reducing the risk of intractability.

- The notion of 'context' can be built into memory, as we shall see in the next chapter.

2.6 An Introduction to Reader-Based Understanding

In the framework of the conduit metaphor, the meaning of a text is placed in the words of the text by its author. As I have already explained, contextual meaning is problematic for this approach and, therefore, it is postulated that text has a *single determinate meaning* that is obtained by a 'competent reader', that is, one who possesses the *correct* set of *a priori* codes of interpretation. In other words, meaning is decontextualized and *objectified*, comprehension is reduced to an invariant algorithm that ignores all *subjective* (or equivalently, *private* or *idiosyncratic*) aspects of the act of interpretation itself.

The idiosyncratic facet of linguistic comprehension is well recognized in psycholinguistics. Mitchell (1982, chapter 7), for example, discusses at length the following individual differences in reading:

- Acquired dislexia.

- Other physiological deficits (*e.g.,* eye movement control, iconic memory).

- Differences in word recognition.

- Differences in phonetic recoding.

- Differences in the use of syntactic and semantic constraints.

- Differences in the accessing or knowledge of word meanings.

- Differences in constructing and combining propositions.

- Differences associated with the interaction of reading subskills (p.173):

> The more fluent readers may be capable of completing some of the subprocesses relatively automatically.... If so, they may be free to devote more processing capacity to other critical aspects of the task.

He concludes his study with the following remark (p.175):

> There are remarkable differences between the levels of skill attained by people with many years of practice in reading.

Thus a normative and self-validating approach to meaning is questionable. Consider, for example, this remark by Northrop Frye (quoted in Hirsch, 1967, p.1):

> It has been said of Boehme that his books are like a picnic to which the author brings the words and the reader the meaning. The remark may have been intended as a sneer at Boehme, but it is an exact description of all works of literary art without exception.

"Among the many developments in literary criticism in the past two decades has been the emergence of a group of German critics, who operate under the banner of *reception theory*, and a less cohesive group of American critics, who operate under the umbrella term *reader response criticism.* Both German and American critics of this persuasion have displayed a shift in concern from the author and the work to the text and the reader" (Plantinga, 1986). Robert Holub (1984, p.149) elaborates:

> The conception of an objective and eternal work of art with a unique structure and a single, determinate meaning was replaced by a variety of models in which the essence of the work is a never-completed unfolding of its effective history, while its meaning is constituted by the interaction between text and reader.

In other words, the *act* of interpretation becomes central; comprehension is taken to proceed from the *private response* of a reader to a text: this constitutes a *reader-based* approach to understanding. For example, Hans-Georg Gadamer (1976) suggests that the act of interpretation be understood as an interaction between the *horizon* provided by the text and the horizon that the interpreter brings to it. Gadamer insists that every reading of a text (whether 'literary' or not) constitutes an act of giving meaning to it through interpretation. From this perspective, the text acts as a *stabilizing* factor in comprehension: the techniques employed by the author (*e.g.*, the spacing of related episodes), *may* constrain the *idiosyncratic* interpretation of a reader. But this comprehender is never seen as an autonomous, idealized individual; "he is neither an abstract phenomenological subject nor an ideal perceiver" (Holub, p.32).

This individualized act of comprehension can be modeled with a trivial algorithm since the qualitative data, that constitutes a reader's private horizon, is completely separated from the algorithm itself. The trivial algorithm merely defines a computational model that encodes the quantitative aspects of comprehension. The algorithm fixes the general form of operations on a fundamental level and thus, affects the quantitative organization and retrieval of the data. But understanding is neither in the algorithm, nor in the text alone, nor in a reader's horizon alone: it results from the interactions between these three entities. It follows that the notion of an 'optimal reading' is abandoned: comprehension is not viewed as a problem for which there exists a (correct) solution, but rather, as the idiosyncratic response of a reader to a text. Each comprehender brings his or her private horizon to the act of interpretation, the text providing a factor of uniformity across responses. Another important factor of uniformity stems from the linguistic and conceptual *conventions* a reader inevitably acquires. In other words, even though all 'knowledge' in **IDIoT**, the proposed comprehension tool, is user-specified, it is highly probable that two distinct users would construct knowledge bases that share a significant number of knowledge units corresponding to the numerous conventions that they were taught, either explicitly or implicitly, as members of a social community. Deciding whether or not these conventions are innate lies beyond the scope of my research. And, since even a well-established convention (*e.g.*, a metaphor but also a grammatical rule), can be modified with time and by usage, I do not claim that the rules and concepts presented in the rest of this work form a corpus of conventionalized knowledge or any sort of set of correct rules for comprehension. The suggested rules and concepts are merely illustrative, and the issue of their 'correctness' must not distract us from the object of study, namely, the *modus operandi* of a quantitative memory for reader-based comprehension, a type of memory that all human brains are assumed to share.

Recapitulating, we have seen that within the conduit metaphor, that underlies the prevailing information-processing paradigm for text comprehension, it is postulated text has a single determinate meaning that is obtained by a competent reader, that is, one who possesses the correct algorithm of interpretation. The existence of the rules, called macrostructures, that form the interpretation algorithm, has been questioned. I have abandoned this approach, in which understanding is reduced to an invariant algorithm, in favor of a reader-based strategy that acknowledges the idiosyncratic aspects of linguistic comprehension. More specifically, I have suggested that the individualized act of comprehension can be modeled with a 'trivial' (*i.e.*, strictly quantitative) algorithm. My task now is to develop a model of memory based on such an algorithm.

Chapter 3

Data Representation and Processing in a Time-Constrained Memory

3.1 Introduction

In the first chapter of this dissertation, I introduced a set of working hypotheses for the development of a model of memory which takes into account the computational constraints imposed by neuronal modeling and can be used for conceptual analysis. The motivations for my choice of human memory as my basic metaphor, of a reader-based approach for text understanding, and of the notion of a 'trivial' algorithm for the specification of my model, have been presented in the preceding chapters. In this chapter, I will discuss the organization and *modus operandi* of my model of time-constrained memory, a quantitative (*i.e.*, non-semantic) model of memory on top of which a conceptual analyzer can be constructed, as will be described in part 2 of this thesis.

I want to emphasize once more that the fundamental postulate of this work is the assumption that linguistic comprehension and its underlying memory processes are time-constrained. Also recall that the mechanisms that shape human memory into an adaptable system are not addressed in this research. Thus, the following set of hypotheses constitutes the starting point for the proposed model of memory:

- Memory is a massively parallel network of simple computing units that exchange simple signals.

- Mechanisms that shape human memory into an adaptable system will not be addressed.

- Memory follows a *trivial algorithm*, that is, a strictly quantitative algorithm that fixes only the general form of memory operations. All memory processes are strictly quantitative.

- All 'knowledge', that is, all qualitative information manipulated by the proposed comprehension tool, is strictly user-specifiable.

- Linguistic comprehension is a teuchistic process that must be defined in terms of quantitative memory processes.

- Of the multitude of possible factors that constrain linguistic comprehension and its underlying processes, the focus is on the importance of quantitative time.

The conceptualization of memory as a massively parallel network of simple computing units exchanging simple signals leads to the following topics to be addressed in the next sections:

1. The organization of memory.

2. The structure and behavior of the computing units of the network.

3. The nature of signals and communication within the network.

Throughout this discussion, the omnipresent role of quantitative time (*i.e.*, time as it pertains to memory management and memory processes such as retrievals) will be emphasized. I will also suggest how the teuchistic process that underlies linguistic comprehension need not be implemented as a global network algorithm but rather as a distributed process, that is, described with respect to the behavior of each computing unit of the network. Also, assuming that all knowledge is strictly user-specifiable requires a tool for the specification of knowledge. Such a tool, called the *knowledge browser*, will be presented in section 3.5.

In the next sections, memory processes will be discussed in isolation of one another. Details of the algorithms and processes of the model will be summarized in section 3.6. In order to clarify the interaction of the memory processes and recapitulate the characteristics of the model, I follow with an annotated trace of the processing of the sentence "John drinks gin". To conclude this chapter, I summarize the different programmable facets of the proposed model of time-constrained memory and briefly discuss some of the insights gained from experiments with the current prototype.

3.2 The Organization of Time-Constrained Memory

As mentioned earlier, much like connectionist researchers, I view memory as a massively parallel network of simple computing units. This functional conceptualization is essential to the proposed model of time-constrained memory, yet incomplete in that it ignores the fundamental feature of conceptual analyzers, namely, the ability to construct 'new' cognitive structures through processing. A construction process in memory suggests an intuitive and commonly accepted dichotomy between *a priori* 'static' knowledge and 'dynamically built' structures, that is, between a static and a dynamic memory. However, since these dynamic structures are ultimately composed of elements of static memory, the dichotomy is tenuous. Indeed, as Hinton and Plaut (1987) suggest, static memory can be thought of as a set of stable 'slow' links between memory elements, whereas dynamic memory could consist of a set of fast, impermanent connections over the same memory elements.

In the current prototype of **IDIoT**, the knowledge specified by the user is completely static, that is, remains unchanged throughout the processing of an input text. The elements of the knowledge base (hereafter KB) are called *knowledge units* (KUs); they form the contents of the static atemporal memory. In contrast, the 'created' structures, which shall be called *clusters*, exist in a separate dynamic memory and cannot be automatically integrated with static memory. This simplified organization could be abandoned in favor of a truly homogeneous model of memory, as will be suggested in chapter 10 of this dissertation.

Dynamic memory must also be organized with respect to time, that is, dynamic memory must be partitioned into temporal stages, as suggested by both neurological evidence (Squire, 1987) and psychological (Baddeley, 1976) evidence. Generally, a short-term memory (STM) and a long-term memory (LTM) are postulated. STM is assumed to have a limited capacity and its elements are taken to be, by definition, more readily accessible than those of LTM. In the simplest form, limits on STM capacity are captured by stipulating its maximum size, though more sophisticated approaches to STM capacity are possible (*e.g.*, Schweickert and Boruff, 1986). An element's belonging to STM is constantly reviewed with respect not only to STM capacity but also to the duration of the membership itself. More specifically, it is commonly assumed that the elements of the STM have a certain 'energy' (or activation) level that quickly (exponentially) decays with time. Once the energy level of a member of STM falls below a certain threshold, it is either 'moved' to LTM or forgotten (Graesser and Clark, 1985). Conversely, elements of LTM decay at a very slow rate over days or years—a process that is irrelevant from a computational perspective.

I adopt this common characterization of memory but, following the ideas of Baddeley (1986), I also identify a working memory (WM), which is the subset of STM whose elements are *'immediately'* and *simultaneously* accessible. In other words, the elements of WM are necessarily accessible. Furthermore, I assume that WM has a very limited capacity. I have adopted the traditional capacity of seven elements for WM.

Clusters 'move' between WM, STM, and LTM. To 'move' does not consist in a physical transfer between partitions of memory, but rather in a change of membership from one partition to another. I assume a cluster is constructed in WM and eventually decays to STM and LTM, if it is not entirely forgotten (*i.e.*, deleted from dynamic memory). Elements of STM all start with the same activation level and the same decay rate. Therefore, for memory management (*i.e.*, enforcement of capacity limits and decay thresholds), it is sufficient to organize the elements according to their time of arrival in STM (not WM): the 'oldest' clusters in STM are also the 'weakest' (with respect to their level of activation), and thus the ones most likely to be 'moved' to LTM during memory management. An ordering is not required in LTM, for which there is no capacity limit or decay, nor in WM, in which, by design, all elements are equally accessible. And since the proposed model of memory is to be strictly quantitative, there is no need for qualitative partitions of memory (*e.g.*, semantic *versus* episodic memory, see Tulving, 1983, 1984).

At any time, a cluster can be *retrieved* from LTM or STM and 'moved' to WM. I postulate that a cluster must be in WM before a construction process can manipulate it. Following Baddeley's (1986, chapter 10) idea of a centralized executive, I hypothesize the existence of a *memory manager* responsible, among other duties, for verifying the membership of a cluster in WM and STM, and for 'transferring' clusters back and forth between WM, STM, and LTM. In **IDIoT**, retrieval is viewed as an atomic operation performed by the memory manager, which *hides* all details of memory management.[1] The focus is placed instead on the notion of *reachability*: given a time-constrained process, that is, a process that must complete its execution before a certain deadline, a cluster can be retrieved by the process if and only if it can be accessed before the deadline. In other words, within the paradigm of a time-constrained memory, the issue is not how a cluster is retrieved, but whether or not it can be reached for retrieval before the execution deadline of the particular process that wants to use it. The notion of reachability only applies to clusters (not KUs) and ties in with the organization of dynamic memory into temporal stages. By definition, all elements of WM are always reachable. For STM, the ordering of elements with respect to their time of arrival defines an ordering for reachability, the 'oldest' clusters being the less reachable ones. This chronological ordering does extend to LTM though, typically, clusters in LTM are seldom reachable (Márkus, 1983).

The temporal partitions of dynamic memory need not correspond to anatomical or physiological separations in the brain, and clusters need not be physically moved from one temporal partition to another. Indeed, as previously mentioned, it merely suffices to have the memory manager keep track of the membership for the WM and STM. More precisely, when the memory manager decides that a certain cluster must be moved from one partition to another, it merely updates its internal membership list for STM and/or WM. For example, when moving a cluster from LTM to WM, the memory manager updates its membership list for WM and stores the time of arrival. Let me summarize the rules for 'moving' between the partitions of dynamic memory:

- A cluster moves from WM to STM when it has sufficiently decayed.

- Upon retrieving a cluster from STM or LTM, if the capacity limit of WM has already been reached, the 'oldest' cluster of WM is moved to STM to 'make room' for the new arrival in WM.

- A cluster moves from STM to LTM when it has sufficiently decay.

- Upon 'receiving' a cluster from WM, if the capacity limit of STM has already been reached, the 'oldest' cluster of STM is moved to LTM to 'make room' for the new

[1] Models of retrieval (or access) for human memory, whether psychological or neurological, can involve complex notions such as those of engram and synergy (see Tulving, 1983; Squire, 1987). Such details lie beyond the scope of this research.

arrival in STM.

- A cluster in STM or LTM may be retrieved, that is, put in WM, if and only if this cluster is reachable by the retrieval process.

- A cluster is said to be forgotten when it is deleted from STM and not moved to LTM.

Clusters are cognitive structures built while processing an input. Without immediately describing their exact nature, let me note that clusters are ultimately 'composed' of elements of the KB. I use the word 'ultimately', for I assume that the 'components' of a cluster can be other clusters, but that, if we think of a cluster as a hierarchical (tree-like) structure, its leaves 'correspond' to KUs. Because KUs permanently reside in static memory, they cannot be, in the current prototype, genuine components of clusters, which are elements of dynamic memory. Therefore, the leaves of a cluster merely refer (or point) to KUs.

3.3 Communication in the Network

Having adopted the conceptualization of the knowledge base as a static memory in the form of a massively parallel network of simple computing units, let us briefly focus on the exchange of signals between KUs. (The clusters of dynamic memory are not computing elements, but merely constructed cognitive structures, and therefore do not exchanges signals. In other words, communication is restricted to static memory.)

Knowledge units communicate through what I shall call input and output 'ports'. The role of time during processing is greatly emphasized by the assumption that, in **IDIoT**, the exchange of a signal between two KUs consumes time. More precisely, if the user specifies that a KU x sends a signal to a KU y, then the user must specify (in the KB) a time delay (or cost) for this exchange. Another (possibly different) delay may be incurred if y sends a signal to x. In other words, communication links are not necessarily symmetric with respect to their time delays. Furthermore, I introduce the intuitive notion of *familiarity* to define an *a priori* static order of 'retrieval', with respect to communication, for the elements of the KB. Each KU needs a user-specified *retrievability coefficient.* If a KU sends a signal to two others, with the same communication delay for both, then the 'most familiar' of the two receivers, that is, the one with the lowest retrievability coefficient, will receive the signal before the other one. More specifically, the *actual* (as opposed to the user-specified) delay of an exchange at time *t* is computed as the product of the *a priori* delay of the exchange with the retrievability coefficient of the receiver at *t*.

Two 'high priority' signals, 'forced activation' and 'forced inhibition' are assumed to have a priority greatly superior to the other signals used in IDIoT, that is, to 'instantaneously' reach their destination (regardless of the retrievability of this destination). 'Instantaneous' communication, that is, communication that requires zero time, is only asymptotically possible and thus actually requires a minimum time quantum, which I call 'epsilon'.

Another fundamental time quantum used in IDIoT is the 'character quantum', that is, the time it takes to read and recognize a character. In IDIoT, the time it takes to read and recognize a word is taken to be equal to the product of the number of characters in this word with the character quantum. Consequently, a text of *n* characters is processed in *n* character quanta, thus emulating real-time processing.

3.4 The Internal Structure and Behavior of Knowledge Units

3.4.1 Knowledge Units As Feature Detectors

Forward Chaining Feature Detection

As mentioned above, KUs are the construction material for the distributed teuchistic process that builds clusters and is taken to underlie the task of linguistic comprehension. Neurological evidence suggests thinking of the elements of the network forming static memory as feature detectors (Squire, 1987), a viewpoint also adopted by connectionist researchers (*e.g.,* Feldman, 1985b). A feature is a qualitative entity (*e.g.,* a phoneme, a sememe, a syntax rule, an inference rule, etc.) and each computing element of the network, that is, each KU, is capable of recognizing several 'configurations (or patterns) of features'. It is left to the user to decide whether a localist representational scheme (which associates one KU to one feature) or a distributed one (which associates several KUs to one feature or several features to one KU) is more adequate. In this dissertation, I have adopted, for both clarity and simplicity, a one-to-one mapping between KUs and features. Consequently, in the rest of this work, these two terms will be used interchangeably.

A commonly accepted conceptualization in both connectionism and psychology (*e.g.,* Norris, 1986) is thinking of a 'configuration of features' as a weighted sum of the features. A *constraint* is formed of such a configuration together with a numeric threshold; the constraint is said to be satisfied once the weighted sum exceeds the associated threshold of detection. From local connectionism (Feldman, 1985b), I adopt the following characteristics:

- Each KU has an input port for each distinct feature used in its constraint.
- The value of a constraint at time t_1 is computed using the value at t_1 of the input port associated with each feature of the constraint.
- Once a constraint is satisfied, its associated computing unit x notifies all other units whose constraints refer to the feature associated with x by sending out a *presence signal* whose 'strength' varies from 0 to 1 and denotes the degree of presence of the feature.

In **IDIoT**, the KUs whose associated feature is referred to in a constraint of a KU x are called the *suppliers* of x; similarly, those that have a constraint that refers to the feature associated with a KU x are called the *customers* of x. It follows that a KU has supplier and customer ports. For now, we will assume that KUs exchange only *presence signals*, that is, signals between 0 and 1 (inclusive) denoting the degree of presence (or detection) of a feature.

I make several enhancements to the local connectionist model; some are discussed in the paragraphs below; others will be introduced throughout this section. I must emphasize that these enhancements merely simplify the data specification task of the user (by providing intuitive labels for certain data), but do not improve the model of memory itself in any way. In particular, no claims are made about the biological plausibility of the proposed enhancements.

First, in order to be capable of detecting several configurations of features, an element of the network, that is, a KU in **IDIoT**, is allowed to have several constraints. The order in which these constraints are specified by a user is important, for it defines the order in which the constraints will be evaluated for satisfaction: the first constraint will be considered before all others, and so on. Once a constraint is satisfied, the KU is said to be *activated* and its associated feature(s) to be *detected*.

Second, I assume that all constraints in static memory are evaluated with respect to a unique detection threshold (across the KB) initially specified by the user. Typically, a higher initial detection threshold will result in fewer features being detected, and thus far less material being available for the construction process.

Third, I distinguish *triggers* and *exceptions* from the other suppliers of a constraint, which are simply called *inputs*. *Triggers* are those features that must be present in order for the constraint to be satisfied. They act as the preconditions of a constraint and therefore, need not be weighted. Triggers can be ordered, in which case presence signals from their corresponding suppliers must be received in the specified order. (A presence signal that is not in order is simply ignored.) A constraint that has received all its triggers is said to be *triggered*. Only the first triggered constraint of a KU can be satisfied, and thus, I repeat, the order in which constraints are specified is extremely important. *Exceptions* are those features of a constraint whose presence decreases the possibility of satisfying the constraint. Exceptions are weighted, as the presence of an exception does not necessarily prevent the satisfaction of a constraint (*e.g.,* the comprehension process manages to parse and understand 'ungrammatical' sentences). As explained below, exceptions also delay the satisfaction process.

Fourth, decay is integrated with constraint satisfaction. More precisely, each supplier port of a KU is taken to hold a queue of presence signals received from its corresponding supplier (see section 3.6). A presence signal starts decaying as soon as it is received. Signals that decay below a certain threshold are considered obsolete and are automatically purged from their queue. The value of a port at time t_1 is computed as the sum of all presence signals stored in its queue at t_1. In other words, constraint satisfaction is implicitly limited by time: all required presence signals must be received within a short interval of time otherwise some will have decayed so much that the constraint cannot be satisfied unless they are received again.

Finally, within the framework of time-constrained memory, feature detection is also made explicitly time-constrained by implementing it as a race process that is given a fixed amount of time to execute. More specifically, once a KU has one of its constraints triggered, it becomes a *candidate* (much as in the model proposed by Norris, 1986) and is given a fixed amount of time to satisfy the constraint. During a candidacy, only the triggered constraint can be satisfied and its sum is recomputed each time a new presence signal from a supplier is received. If the triggered constraint has exceptions, then the sum is checked and satisfaction is possible only at the end of the candidacy's delay, in order to allow any signal from an exception sufficient time to reach the candidate KU. Otherwise, satisfaction is possible at any time within the interval of candidacy. At the end of the candidacy, regardless of success

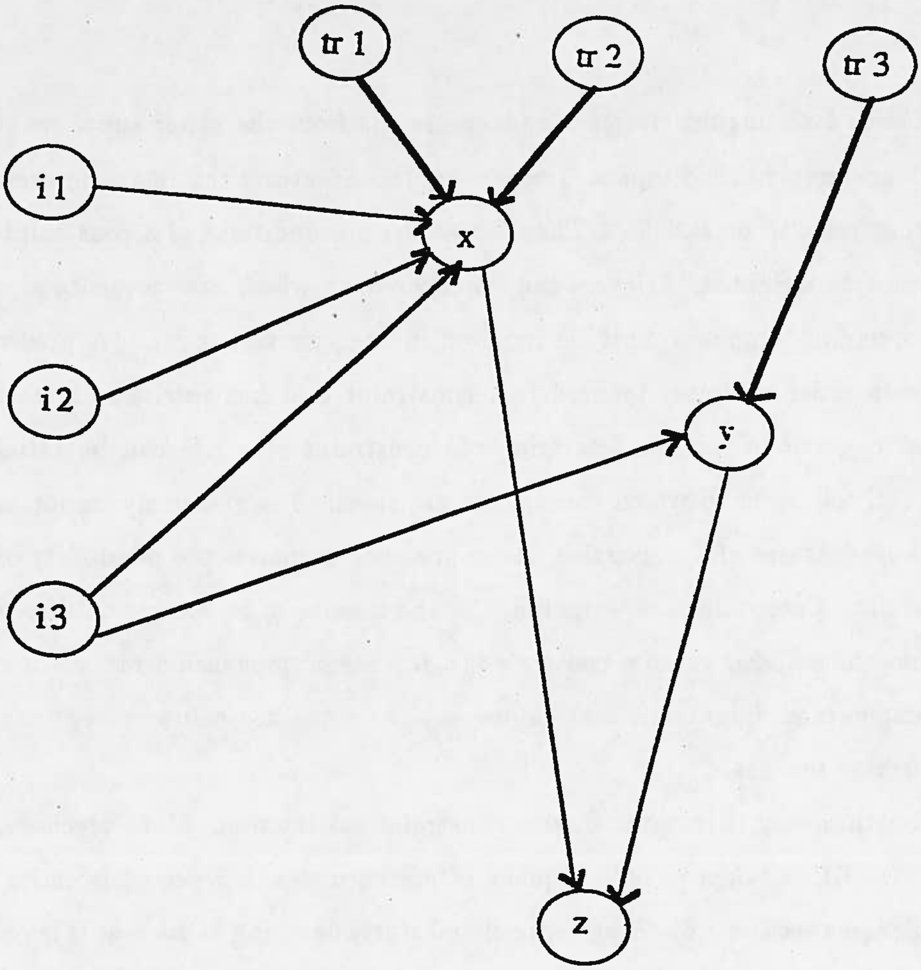


Figure 3.1: A simple example

(*i.e.*, satisfaction) or failure (*i.e.*, the sum of the triggered constraint is below the detection threshold), a KU resets all its constraints by emptying the queues of its supplier ports.

We are now ready to consider a simple example illustrated in figure 3.1. The notation is that of the implementation.

```
KU x:
 constraint c1:
  triggers: tr1, tr2
  inputs:
      i1 has a weight of 0.9
      i2 has a weight of 0.1
 constraint c2:
  triggers: tr1, tr2
  inputs:
      i1 has a weight of 0.5
      i3 has a weight of 0.5

KU y:
 constraint c1:
  triggers: tr3
  inputs:
      i3 has a weight of 1

KU z:
 constraint c1:
  inputs:
      x has a weight of 0.5
      y has a weight of 0.5
```

KU x has two constraints, each one becoming triggered as soon as it has received a presence signal from both KUs tr_1 and tr_2 (in any order). Constraint c_1 of x has two inputs i_1 and i_2. In the current prototype of **IDIoT**, the sum of the weights of the inputs of a constraint must be less than or equal to 1. Intuitively, the weight of an input defines its relative 'necessity' with respect to the current detection threshold. In c_2 of x, the inputs carry the same weight, and thus are equally 'important' for the satisfaction of the constraint. Conversely, in c_1 of x, i_1 is far more needed than i_2; with a detection threshold at 80 percent of the maximum detection threshold, c_1 could be satisfied with only a presence signal from i_1.

If x and y receive a presence signal on all of their trigger ports and on all their input ports, then they become activated and send a presence signal to z that, in turn, becomes activated. Because of the ordering of its constraints, x becomes activated through the satisfaction of its constraint c_1.

Backward Chaining Feature Detection

The process of constraint satisfaction described so far requires only one signal, namely, the presence signal (whose value varies from 0 to 1), but only allows for the forward chaining of feature detections: a constraint cannot be satisfied and, thus, a feature cannot be detected unless its suppliers have been activated. I now introduce the notions of *submission*, *confirmation*, and *reinforcement*, in order to accommodate the backward chaining of feature detections. Intuitively, as soon as a feature has one of its constraints triggered (*i.e.*, satisfies one of its sets of preconditions), it need not wait for signals from its inputs, but rather it may immediately ask the latter for 'positive feedback', that is, for some confirmation of its 'felicity' if it were activated. More specifically, I propose that when at time t_1 a KU x becomes a candidate, it verifies the value of all the supplier ports referred to in its triggered constraint. A missing feature is one whose corresponding port in x has no queue of signals at t_1. A ***submission signal*** is sent by x to all the missing features of its triggered constraint. This signal acts as a conditional presence of x. Upon receiving such a signal, any KU y associated with a missing feature of x treats the signal as a maximum presence signal (*i.e.,* the value 1) from x and checks whether one of its constraints would be satisfied by this input. If one would be satisfied, then y sends back a confirmation signal to x and starts waiting (for a short fixed amount of time) for a reinforcement signal from x; but the feature associated with y does not become detected. If, on the other hand, y has one of its constraints triggered but not satisfied by the submission signal from x, then y, in turn, sends a submission signal to the missing features of its triggered constraint. In all other cases, the submission signal to y is ignored. Each KU z that is not the originating candidate and that sends a submission signal to the set of its missing features waits for a fixed amount of time for a confirmation signal from each member of this set. Once z receives all the required confirmations, it sends a confirmation signal to the KU from which it received a submission signal. If the candidate x receives a confirmation signal from all its missing features before the deadline of its candidacy, its triggered constraint is considered satisfied and the feature associated with x is detected. Intuitively, x has received enough 'positive feedback', *confirmation* of its felicity if it were activated. Upon its detection, x not only notifies its customers, but also sends a *reinforcement signal* to its missing features. Upon the reception of this reinforcement signal, the missing features first relay this reinforcement signal to the KUs that sent them confirmation signals, and then become themselves detected.

This process of submission/confirmation/reinforcement is somewhat similar to the spreading activation mechanisms of marker-passing models (*e.g.,* Hendler, 1989; Norvig, 1989) but presents some advantages. Since only the missing features of would-be triggered constraints are used to relay submission signals, the number of possible chains or paths (of submission/confirmation signals) is far more constrained than in marker-passing systems, and, by design, a path of confirmation signals is always a 'useful' path. Furthermore, there is no need for an often arbitrary or intractable mechanism for evaluating paths: for each missing feature, the path (of confirmation signals) that 'wins' is simply the first one to send a confirmation signal back to the originator of the initial submission signal. Since the proposed scheme is time-constrained, the only restriction for a candidacy is that all required confirmation signals arrive at the initiating candidate before the deadline of the candidacy is reached Also, the proposed scheme does not use complex markers, but only three numeric signals (for submission, confirmation, and reinforcement). And, finally, the submission/confirmation/reinforcement mechanism does not impose restrictions on the lengths of paths and captures the intuition that the most 'reconstructable' knowledge is not necessarily put in dynamic memory. Let me explain. In **IDIoT**, contrary to typical marker-passing systems, a path of features does not become activated as a whole but, rather, is incrementally constructed, through reinforcement signals, starting with the originating candidate x. Given a path of confirmation signals between a candidate x and a KU y (that sent back the first confirmation signal of the path), y waits a fixed amount of time for a reinforcement signal from the KU to which y sent a confirmation signal. Again, since communication consumes time, the longer the path (in terms of KUs) between x and y is, the longer the confirmation signal originated by y will take to be relayed to x, and thus provided x becomes activated, the longer a reinforcement signal originated by x will take to be relayed to y. It is therefore possible that y (as well as some of the KUs that precede it in the path between x and y) will not receive the reinforcement signal it is waiting for before its waiting race expires, that is, before y stops 'listening' for the signal. It follows that it is possible that only a few of the KUs forming the path between x and y become activated. But this is not a problem for, given the same 'context' (*i.e.*, the same contents of dynamic memory and the same parameters of memory), the path can be reconstructed through the same process. Also, the more of the path that is already detected when attempting the reconstruction, the faster the missing features will be detected since, in essence, part of the construction is already present.

To clarify this discussion, let us consider a simple exchange of signals (illustrated in figure 3.2):

1. In the sentence "John eats caviar", the word 'caviar' is recognized and triggers the candidacy of the feature **caviar** associated with the concept of caviar.

2. Feature **caviar** submits (*i.e.*, sends a submission signal to) its only missing feature **noun**.

3. Feature **noun** relays the submission signal it receives from **caviar** to its customer

NP.

4. Feature **NP** (noun phrase) relays the submission signal it receives from **noun** to its customers including **directObject**, **indirectObject**, **PP**, etc.

5. Feature **directObject** had its constraint triggered earlier by the detection of a subject-verb relationship satisfied by the submission signal received from **NP**. In essence, an NP following a verb is treated as the initial direct object. Feature **directObject** does not become detected, but instead sends a confirmation signal back to **NP**.

6. Feature **NP** relays the confirmation signal it receives from **directObject** to **noun**.

7. Feature **noun** relays the confirmation signal to **caviar**.

8. Having received a confirmation signal from all its missing features, **caviar** becomes detected. It sends a presence signal to its customer **food** and a reinforcement signal to **noun**.

9. Feature **noun** receives the reinforcement signal of **caviar**, becomes detected, and sends a reinforcement signal to **NP**.

10. Feature **NP** receives the reinforcement signal of **noun**, becomes detected and sends a reinforcement signal to **directObject**.

11. Feature **directObject** receives the reinforcement signal of **NP** and becomes detected. It sends a presence signal to its customers.

If the candidacy of **caviar** expires before it receives a confirmation signal from **noun**, then the path between **caviar** and **directObject** is not used and the candidacy fails. If, while waiting for reinforcement, **noun**, **NP**, or **directObject** have their time expire, then they will not become activated and the reinforcement signal that will eventually reach them will be ignored. If we assume that this was the case for **directObject**, then given the same context, that is, the prior detection of a subject-verb relationship and the detection of **caviar** and **noun** and **NP**, **directObject** is readily reconstructable as its constraint can be directly triggered and satisfied.

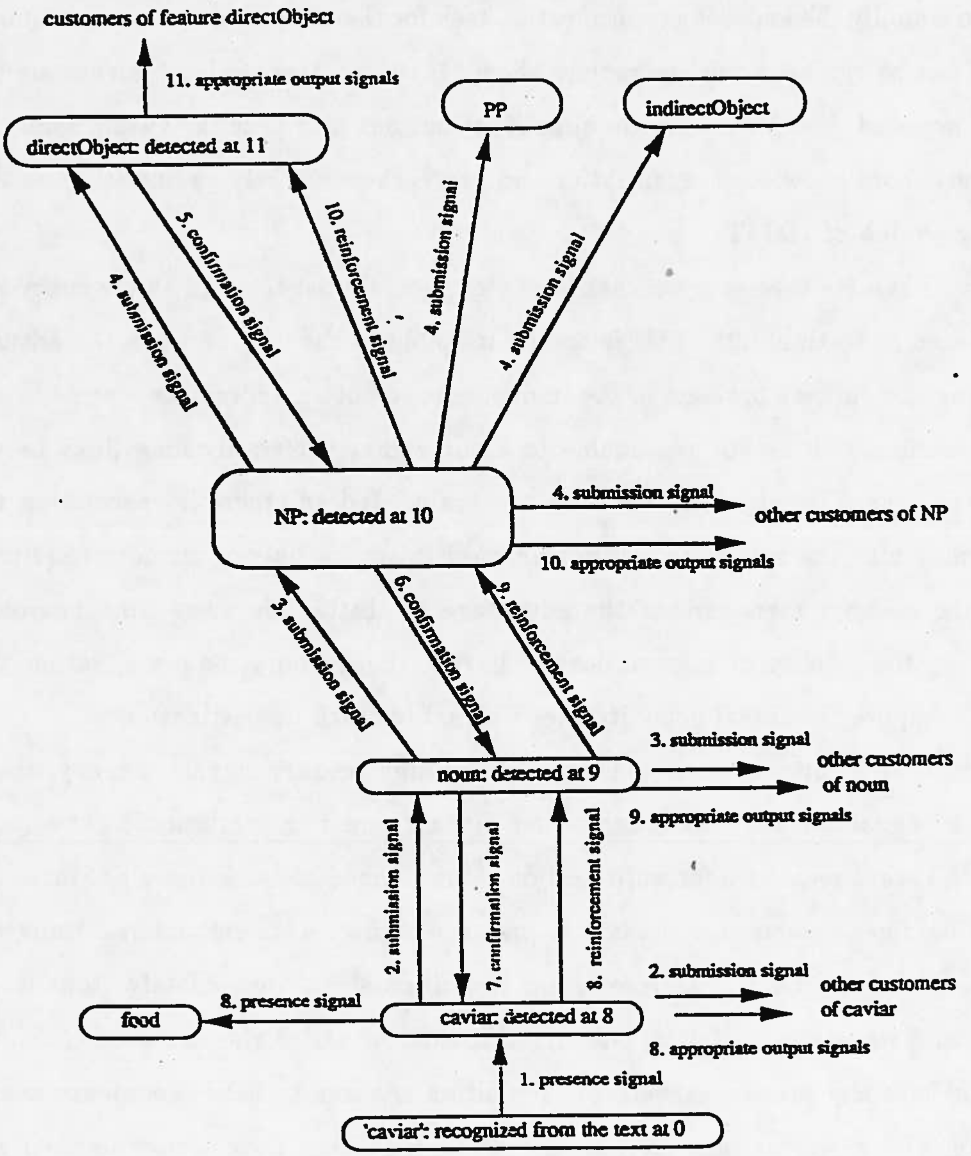


Figure 3.2: A simple exchange of signals

Other Enhancements

The race processes and constraints briefly described in the previous sections form the essence of the internal organization and behavior of KUs. Several other enhancements to the conceptualization of KUs as feature detectors are made in the current prototype of **IDIoT** in order to simplify the knowledge specification task for the user. None are truly required since 'tricks' can be typically used to replace them. However, these enhancements are included in the proposed model not only for simplification, but also to make visible some common intuitions about knowledge organization and use, rather than relying implicitly on the exact *modus operandi* of **IDIoT**.

First, I assume that every exchange of signals is routed through the memory manager (introduced in section 3.2). This is totally invisible to the user and has the advantage of bypassing the difficult problem of the implementation of an addressing scheme in memory. More specifically, it is not reasonable to assume that arbitrarily long links between all communicating KUs physically exist in the brain. Indeed, there is speculation that the cortex may play the role of an addressing mechanism for human memory (Squire, 1987). Using the memory manager has the advantage of abstracting away from neurology and preserving the locality of information by having the memory manager, rather then the sender, compute the actual delay it takes a signal to reach its destination.

Second, as mentioned earlier, there are two high priority signals, namely, the 'forced detection' signal and the 'forced inhibition' signal (hereafter 'inhibition'). At any point in time, a KU that receives a forced detection signal immediately becomes activated (*i.e.,* its feature becomes detected) and sends a presence signal to its customers. Conversely, at any point in time, a KU that receives an inhibition signal immediately stops its current process and reverts to its initial mode, which shall be called the idle mode (*i.e.,* the KU does not have any process executing). Inhibition is taken to have precedence over forced detection. Both signals take epsilon time to be sent from their sender to their receiver, regardless of the retrievability of the receiver and of the *a priori* delay between both KUs. This is made possible by the previous assumption that all signals are routed through the memory manager (which simply assigns an epsilon delay to these high priority messages). Furthermore, in order to be instantaneously processed by their destinations, these signals are received on a special input port, called the *manager port*, that every KU has. As soon as a high priority signal is received on this port, it is immediately processed, regardless of the state of the KU at that point in time. And, for convenience, I allow the user of **IDIoT** to distinguish *associations* from the other customers of a KU. The associations of a KU are those customers to which it sends a forced detection signal upon its activation. For example, the feature **food** could be an association of **caviar** since caviar is necessarily an instance of the concept food. Although not absolutely necessary, associations have been found to be very helpful for implementing a generalization hierarchy, as illustrated by the examples of word sense disambiguation shown in chapter 7.

Third, in order to account for the important phenomenon of expectations (which is akin to priming, see Tulving, 1983), I introduce one last special signal, the *expectation signal*. I view expectations as a mechanism for the speed-up of the detection of a feature. In other words, a feature that is expected will become activated more readily than a non-expected feature. Within the framework of time-constrained memory, I suggest that, for a short fixed amount of time, the receiver of an expectation signal need not satisfy any of its constraints, but rather merely obtain all triggers for one of its constraints in order to become detected. Intuitively, receiving an expectation signal guarantees the felicity of the receiver, which therefore, merely has to assemble its preconditions. Let us consider a simple example:

Example 3.4.1 *John eats at Maxim's. He orders caviar.*

Let us assume that the feature **actionDine** (associated, among others, with the action of eating at a restaurant) sends, upon its detection, an expectation signal to the feature **orderFood**. Then:

1. Having the verb **actionEat** with a well-known restaurant as a complement of location causes the eventual detection of the feature **actionDine**, which sends an expectation signal to **orderFood**.

2. Feature **orderFood** receives an expectation signal.

3. The word 'orders' triggers the feature **orderFood**, which immediately becomes detected, as it has received an expectation signal and merely needs to have one of its constraints triggered.

As a fourth enhancement, I propose associating an *output strategy*, which can possibly be empty, with each constraint of a KU. So far, upon its detection, a KU sends a forced detection signal to its associations and a presence signal to its other customers. The user must specify for each non-association customer a default signal to send. The enhancement consists in allowing the user to override the default output to be sent to a particular non-association customer of a KU x if a particular constraint c of x becomes satisfied. This overriding is achieved by associating a set of (*customer—signal*) pairs to one or more of the constraints of x. Upon its detection, x checks which of its constraints was satisfied. If this constraint has an associated output strategy, then, for each customer referred to in the output strategy of the satisfied constraint, x sends the output signal specified by the output strategy rather than the default signal for this customer. This enhancement has been found to be extremely useful for a feature x in which a customer should be sent an inhibition signal or an expectation signal for only a few of the constraints of x.

Finally, I allow the user of **IDIoT** to specify the *candidates* of a KU. Recall that in the submission/confirmation/reinforcement mechanism introduced above, the chaining process is initiated by a candidate missing one or more features in its triggered constraint, that is, in the constraint the candidate must satisfy in order to become detected. In this process a submission signal is not sent to the customers of the candidate for the simple reason that receiving a confirmation from a feature that would become detected if feature f were active does not entail that f ought to become detected. In other words, a confirmation from a customer is not sufficient evidence to cause the detection of a feature if it has a triggered constraint to satisfy. But what if the triggered constraint has only triggers? In this case, the constraint is satisfied as soon as all triggers are received. If this is generally acceptable, there may be features for which triggers are not enough and a confirmation (necessarily from a customer, as there are no missing features to the triggered constraint) is required. Such features are called the *candidates* of their trigger(s).[2] If KU x has KU y declared as a candidate, then when x becomes detected, it sends a confirmation signal to y. Once y has all its triggers, it sends a submission signal to each of its customers and waits for a fixed amount of time for a confirmation signal from one of them. The KU y becomes active as soon as such a confirmation signal is received. The customers of y act exactly like any KU that receives a submission signal. In other words, the chaining process is the same as

[2] Anticipating on the next part, I remark that the notion of candidates is especially intuitive and useful for word sense disambiguation. For example, the two KUs alcoholicBeverageGin and cardGameGin are candidates of their sole trigger, the KU gin. The presence of gin is not sufficient to have either of these candidates become detected. Instead, each must receive a confirmation from one of its customers. Candidates have also been used as a possible approach to prepositional phrase attachment.

for backward chaining, only how this chaining is initiated differs: whereas, for backward chaining, the candidate is a KU sending a submission signal to its missing features (in order to satisfy its triggered constraint), in the case of an explicit candidate x, x sends a submission signal to each of its customers as soon as it has received a confirmation signal from all its triggers, and x becomes detected as soon as it receives a confirmation signal from one of its customers.

These enhancements will be illustrated at the end of this chapter, as well as in part 2.

3.4.2 Knowledge Units as Cluster Builders

Again, let me recall that KUs are the construction material for the distributed teuchistic process that builds clusters and is taken to underlie the task of linguistic comprehension. The set of clusters constructed during the reading of the text constitutes the output of **IDIoT**, the interpretation that the user can examine. In this subsection, I focus on the process of the construction of clusters.

Organization of a Cluster

Let me start by elaborating on the nature of clusters. Recall that the proposed model of memory is strictly quantitative, that is, deprived of any qualitative information. Clusters must be general enough to allow the user to construct almost any kind of representation (*e.g.*, parse trees, scripts, etc.) that is hypothesized for comprehension. Of the multitude of possible organizations for clusters, I have selected a very simple one: a cluster is defined as a non-empty set of features, each feature *governing* a (possibly empty) set of clusters. In other words, a cluster is a hierarchical structure whose 'leaves' are features. Consider, for example, the cluster associated with the noun phrase "a park":

```
cluster <identifier1> with features:
 KU NP
 KU NPhead that governs:
  cluster <identifier2> with features:
   KU 'park'
   KU singular
   KU park
   KU noun
   KU 3rdPerson
KU NPquant that governs:
 cluster <identifier3> with features:
  KU determiner
  KU singular
  KU adjectives
```

I emphasize that the contents of this cluster are assembled from the knowledge specified by the user, as shall be explained shortly. The 'correctness' or 'completeness' of a representation is not the object of study in this dissertation.

The clusters governed by a feature are called its *subclusters*. Also, each cluster has an identifier, that is, a unique address in dynamic memory, given in the form of a number, and can be shared as a subcluster by other clusters, as in the actual structure produced for the sentence "John watched the rabbit in the park", in which the prepositional phrase "in the park" is taken to be ambiguous and is therefore attached to both the verb "watched" and the noun "rabbit":

```
cluster 16436 in working memory with features:
 KU newFact that governs:
   cluster 17620 in working memory with features:
   KU VP
   KU mainVerb that governs:
     cluster 15957 in working memory with features:
     KU 'watch'
     KU pastForm
     KU actionObserve
     KU verb
     KU activeMood
     KU verbComplemented
     KU location that governs:
       cluster 18999 in working memory with features:
       KU NP
       KU NPhead that governs:
         cluster 16207 in working memory with features:
         KU 'park'
         KU singular
         KU park
         KU noun
         KU 3rdPerson
       KU NPquant that governs:
         cluster 7987 in STM with features:
```

```
      KU determiner
    KU NPmods
KU VPmods that governs:
  cluster 15005 in STM with features:
  KU ‘close’
  KU adverb
KU subject that governs:
  cluster 10935 in STM with features:
  KU NP
  KU NPhead that governs:
    cluster 17028 in STM with features:
    KU singular
    KU ‘John’
    KU proper
    KU person
    KU male
    KU rational
    KU animate
    KU 3rdPerson
  KU NPquant
  KU NPmods
KU directObject that governs:
  cluster 16183 in working memory with features:
  KU NP
  KU NPhead that governs:
    cluster 15035 in working memory with features:
    KU ‘rabbit’
    KU singular
    KU noun
    KU rabbit
    KU animate
    KU 3rdPerson
    KU location that governs:
      cluster 18999
  KU NPquant that governs:
    cluster 17872 in STM with features:
    KU determiner
    KU singular
  KU NPmods
```

In this structure, the cluster 18999 corresponding to the prepositional phrase “in the park” is a subcluster of feature location in both cluster 15957 corresponding to the main verb and in the cluster 16183 corresponding to the direct object. The reader will also notice that the verb ‘to watch’ has been interpreted as ‘to observe’ with the modifier ‘close’ and in the cluster 16183 corresponding to the direct object.

The levels of indentation of the structure correspond to the level of nesting of clusters. The structure consists of one cluster (16436) having only the feature **newFact**, which corresponds to the whole sentence. This feature governs one cluster (17620), which has the four features **VP**, **mainVerb**, **VPmods** (for the modifiers of the VP), **subject** and **directObject**. Feature **mainVerb** governs cluster 15957, whose features mainly capture the fact that the main verb is **actionObserve** in the active mood and that this verb has a location specified in cluster 18999. Feature **subject** governs cluster 10935, which corresponds to the NP "John". The feature **NPquant** is used to hold a determiner, if any. Feature **NPmods** keeps a list of the adjectives of the head of the NP. The knowledge inferred from the feature **John**, namely that this is a male rational animate, has been added to the cluster for the head of the NP. Similarly, feature **directObject** governs cluster 16183, whose features capture the fact that the object being observed by the subject is a rabbit in a park. Finally, the trace indicates the membership of each cluster in one of the temporal partitions of dynamic memory.

Expansion Procedures

Having briefly shown what a cluster looks like, I will now turn to the problem of cluster construction. I suggest that once a KU becomes activated, it asks the memory manager to execute its *expansion procedure*. An expansion procedure is an ordered sequence of cluster operations that modifies the current contents of the dynamic memory. The cluster operations, which are introduced below, provide the basic functionality required for generic data structures: access, addition, deletion, comparison, traversal, etc. A KU does not execute its expansion procedure itself, for two reasons. First, having each KU able to execute cluster operations would violate the assumption that KUs are simple computing units. Instead, the use of a memory manager hides the details of complex processes such as cluster retrieval (*e.g.*, with respect to memory management) within a single abstract entity. Second, having a single entity be able to execute expansion procedures avoids the complex problems associated with several KUs simultaneously modifying the contents of memory. The memory manager executes a single expansion procedure at a time, in the order in which the corresponding KUs become activated. If two KUs are activated simultaneously, the memory manager executes their respective expansion procedures in a random order. (The parallel execution of expansion procedures is possible, in theory, provided 'critical' mutually exclusive segments of procedures can be identified.)

Cluster operations do not refer to the actual addresses in memory of features and clusters. For features, the procedure refers to the name of the corresponding KU. And in order for two instructions of a procedure to refer to the same cluster, the supplied retrieval operations bind the retrieved cluster to a ***user variable***, which consists of a short alphanumeric string. As a whole, an expansion procedure is given a fixed amount of time to execute, and either succeeds or fails. An expansion procedure fails as soon as any of its operations fails and succeeds if none do.

I will now present a list of these operations. I use the following notation: 1) an item between square brackets is optional, 2) (feature) is a feature name, and 3) (uv1) and (uv2) are user variables. Here are the operations:

- addFeature (feature) to (uv1)
  Given the cluster uv1, add the specified feature to it. As a side effect, a forced detection signal is sent to the feature to ensure that it becomes detected, if it is not already.

- addFeaturesOf (uv1) to (uv2)
  The features of cluster uv1 are added to the features of cluster uv2.

- addFirstSubCluster (uv1) to (feature) in (uv2)
  Given the cluster uv2 with the specified feature, the cluster uv1 is added to the set of clusters governed uv2. In fact, a feature governs a collection of subclusters that defines an order of traversal. This instruction makes uv1 the first element of the ordered collection of clusters governed by the specified feature.

- addLastSubCluster (uv1) to (feature) in (uv2)
  Analogous to the previous operation.

- exit
  Terminate with success the expansion procedure.

- findExactReference (uv1) to (uv2)
  Search dynamic memory for a cluster with the exact same set of features and governed subclusters as uv2. If found, this cluster is bound to uv1. FindExactReference fails if

multiple reachable referents are found during its time-span or if no reachable referent can be found within the time-span allocated to its expansion procedure.

- findInclusiveReference (uv1) to (uv2)
  Search dynamic memory for a cluster with a set of features which includes all features and governed subclusters of uv2. If found, this cluster is bound to uv1. Failure conditions are the same as for findExactReference.

- getCluster (uv1) governedBy (feature) in (uv2)
  Search dynamic memory for the first cluster that is a subcluster of the given feature in uv2.

- getCluster (uv1) governing (feature) [in (uv2)]
  Search dynamic memory for the first cluster that governs the given feature. If the optional part of the operation is specified then the search is limited to the subclusters of uv2.

- getNewCluster [(uv1)]
  Create an empty new cluster in WM and bind it to uv1. If no argument is specified, then the predefined user variable 'newCluster' is used.

- moveFeature (feature) from (uv1) to (uv2)
  Move the specified feature and its subclusters from cluster uv1 to cluster uv2.

- moveSubClustersFrom (feature1) to (feature2) in (uv1)
  Move the subclusters of feature1 to become those of feature2 in cluster uv1.

- removeCluster (uv1)
  Remove the cluster uv1, without its subclusters, from dynamic memory.

- removeFeature (feature) in (uv1)
  First delete all subclusters governed by the specified feature and then delete the feature from the cluster.

- renameFeature (feature1) to (feature2) in (uv1)
  Self-explanatory.

- substituteCluster (uv1) for (uv2)

  Cluster uv1 is given the dynamic memory address of cluster uv2 which is then removed.

- testAbsenceOf (feature) in (uv1)

  Succeeds if the specified feature is not in uv1.

- testDifferenceOf (uv1) and (uv2)

  Fails if uv1 and uv2 recursively have the same features and the same subclusters.

- testEquivalenceOf (uv1) and (uv2)

  Succeeds if uv1 and uv2 recursively have the same features and same subclusters.

- testPresenceOf (feature) in (uv1)

  Succeeds if the specified feature is in uv1.

A great deal of flexibility is introduced in expansion procedures by having conditional instructions of the form:

if (constraint name) then (cluster operation)

In such an instruction, the specified cluster operation is attempted only if the specified constraint was the one whose satisfaction led to the activation of the KU that owns the executing expansion procedure. In other words, within the same executing expansion procedure of a KU *x*, different cluster operations can be attempted according to which constraint of *x* was satisfied.

The traversal of a cluster is performed breadth first, that is, each level of a cluster is visited completely before the next one is accessed. Each level costs increasingly more time to access, and thus, the deeper the structure, the more time it takes to reach the 'lower' levels. Therefore, the time it takes to reach a cluster is determined by which partition of dynamic memory it resides in and by the number of levels that have to be traversed in order to reach it. Given that any expansion procedure is allocated a fixed amount of time to execute, a getCluster operation fails if it cannot find a reachable satisfactory cluster in time.

Finally, memory management is made completely invisible to the user of **IDIoT** by being encapsulated in the operations getNewCluster, getCluster, and removeCluster. In its present form, the memory management algorithm implements a simplistic first-in first-out storage/retrieval system for dynamic memory. Enhancements to the management of memory are discussed chapter 10. For now, the point to be grasped is that, most importantly, all the complexities of the *modus operandi* of time-constrained memory (*e.g.*, the complex implementation of the findInclusiveReference operation) are completely hidden from the user of **IDIoT** who specifies KUs without regard to this *modus operandi*. The effect is that users believe that **IDIoT** is a simple, if not trivial, access system to the rules and concepts they specify. This is a most desirable illusion, because it favors the specification of knowledge completely independently from the exact workings of time-constrained memory. But it is also very important to understand that this illusion is just that, an illusion, for the clusters built during comprehension do not proceed solely from the knowledge base, but also depend on the parameters of memory (*e.g.*, race deadlines, decay factor) and on the context, that is, on the contents of dynamic memory at a given point in the reading. In other words, the interpretation of a text results from the interactions between the knowledge base and the time-constrained memory.

Up to this point in the discussion of time-constrained memory, constraints have been specified by the user and depend on receiving signals from suppliers. In order to have the detection of a KU depend alternatively on the contents of the dynamic memory, I introduce the notion of a *buildable feature*. A buildable feature is a feature whose expansion procedure acts as a dynamic constraint that can be satisfied within a short time interval with respect to the contents of dynamic memory. If the expansion procedure of a KU does not include a test operation, then the KU is not buildable, and an error is reported to the user if its expansion procedure fails. The constraints of a buildable feature have only triggers and its expansion procedure must include at least one of the following test operations: findExactReference, findInclusiveReference, testAbsence, testDifference, testEquivalence, testPresence. Once a constraint of a buildable feature x has received a presence signal for all its triggers, x does *not* become activated, but rather attempts, for a fixed amount of time, to have its expansion procedure succeed. Within the time interval allocated by x, the memory manager attempts the execution of the expansion procedure of x each time a new feature is detected, until either the procedure succeeds or time runs out. If the expansion procedure succeeds, x immediately becomes activated; if it fails, nothing happens, and if the deadline is reached, then x reverts to idle mode.

Two other types of special KUs are now introduced. First, *innate features* are KUs that are not buildable and do not have any constraints. These KUs become activated by the presence of a string in the input text. Intuitively, the innate features are the recognizable (*i.e.*, known *a priori*) words of the KB. Second, *relays* are KUs that have no constraints and no expansion procedure, just supplier and customer ports. Their role is limited to relaying the signal received from a supplier to all their customers. A relay groups a set of features under a common name that can be referred by a constraint, thus avoiding specifying a constraint for each member of the group. In the current prototype, it is left to the user to specify whether a KU is innate or a relay.[3]

3.4.3 Examples of Knowledge Units

Here are a few examples of definitions for KUs. A line starting with a % is a comment and the ellipsis (...) is used to omit irrelevant details:

```
innate KU 'caviar':
% This innate feature recognizes the word 'caviar'
% and constructs a cluster with that feature for it.
 associations: caviar
 expansion:
% Use the built-in user variable called 'newCluster' which
% is set by the getNewCluster instruction with no argument.
      getNewCluster
      addFeature 'caviar' to newCluster

KU caviar:
% Food is a generalization of caviar.
% The concept of caviar is triggered by the word
% 'caviar'. The constraint uses input noun to
% enforce the felicity of a noun at that point
% in the processing of the sentence.
% If the concept is detected, then the cluster that
% includes 'caviar' is updated to reflect the new
% information.
 associations: food
 constraint 'caviar':
  triggers: 'caviar'
  inputs: noun has a weight of 1
 expansion:
```

[3] In the future, however, it is conceivable that the implementation of IDIoT could identify such KUs by itself.

```
     getCluster u1 governing 'caviar'
     addFeature caviar to u1
     addFeature noun to u1
 ports:
     'caviar' has a delay of 1 and is a supplier
     noun has a delay of 1 and is a supplier

relay KU noun:
% This relay has all nouns as suppliers and
% only NP as a customer. All rules concerned with a
% noun refer to this relay rather than to each noun.
 ports:
     bench has a delay of 1 and is a supplier
     box has a delay of 1 and is a supplier
     gin has a delay of 1 and is a supplier
     ...
     NP has a delay of 1 and is a customer

innate KU 'ate':
% The word 'ate' is associated with the verb eat.
 associations: actionEat
 expansion:
     getNewCluster u1
     addFeature 'eat' to u1
     addFeature pastForm to u1
 ports:
     pastForm has a delay of 1 and is a customer

KU actionEat:
% Most of the constraints of actionEat capture the different
% morphological forms of this concept.
 constraint 'ate':
  triggers: 'ate'
  inputs: verb has a weight of 1
 constraint 'eats':
  triggers: 'eats'
  inputs:
     verb has a weight of 1
     ...
% However, the concept can be activated by other KUs that
% denote more specialized actions as 'to dine' which has feature
% restaurant as an expectation.
 constraint dine:
  triggers: actionDine
  outputs:
     restaurant is sent signal expectationSignal
 expansion:
% We use conditional instructions to handle the feature actionDine.
% The next 3 instructions are executed only if constraint dine
% was satisfied. In this case, the concept actionEat is added
% to the word 'dine'.
      ifConstraint dine then getCluster u1 governing 'dine'
      ifConstraint dine then addFeature actionEat to u1
      ifConstraint dine then exit
% If we are not dealing with actionDine then we want to associate
% the concept actionEat with the word 'eat' and mark it as a verb.
      getCluster u1 governing 'eat'
      addFeature actionEat to u1
      addFeature verb to u1
 ports:
      'ate' has a delay of 1 and is a supplier
      'eats' has a delay of 1 and is a supplier
      ...
      actionDine has a delay of 1 and is a supplier
      verb has a delay of 1 and is a supplier
      eatFood has a delay of 1 and is a customer

KU missingDirectObject:
% Once at the end of a sentence, detect whether an obligatory transitive
% verb is missing its directObject.  If so, flag a syntactic conflict.
% This feature captures a dynamic rule and thus is buildable as indicated
% by the following line that is automatically generated by the system.
is buildable:
% syntacticConflict is a feature used to capture all
% syntactic violations.
 associations: syntacticConflict
 constraint c1:
  ordered triggers:
      compulsoryTransitive endOfSentence
 expansion:
      getCluster u1 governing newFact
      getCluster u2 governing mainVerb in u1
      getCluster u3 governedBy mainVerb in u2
      testPresenceOf compulsoryTransitive in u3
      getCluster u2 governing VP in u1
      testAbsenceOf directObject in u2
 ports:
      compulsoryTransitive has a delay of 1 and is a supplier
      endOfSentence has a delay of 1 and is a supplier

KU alcoholicBeverage:
% This is another example of conditional instructions. More
% importantly, it illustrates the organization of a is-a
% hierarchy of concepts: the parent is triggered by its children.
% This organization of communication links must not be
% conflated with issues of retrieval. In particular,
% this organization DOES NOT imply that a parent can readily
% enumerate all its children.
 associations: aboutAlcoholicBeverages beverage
 constraint gin:
  triggers: alcoholicBeverageGin
 constraint scotch:
  triggers: alcoholicBeverageScotch
 expansion:
      ifConstraint gin then getCluster u1 governing alcoholicBeverageGin
      ifConstraint scotch then getCluster u1 governing alcoholicBeverageScotch
      addFeature alcoholicBeverage to u1
 ports:
      alcoholicBeverageGin has a delay of 1 and is a supplier
      alcoholicBeverageScotch has a delay of 1 and is a supplier
```

3.5 The Specification Tools

Let us now briefly consider the current prototype of **IDIoT**, implemented in Smalltalk-80 version 2.3.[4]

Recall that all knowledge in **IDIoT** is user-specifiable. To input this knowledge, three different *browsers* (in the Smalltalk-80 sense of this word, see Goldberg, 1984) are used for this task. Roughly put, a browser is a menu-driven interactive window allowing for both the inspection and modification of information. I remark that both the choice of Smalltalk-80 as the language of implementation for **IDIoT**, and of browsers for the specification of knowledge, proceed from the intuitive object-oriented perspective suggested by the parallel distributed nature of the network in which the computing units are the main object of study.

The *knowledge base browser* allows the user to create a KU and assign it a retrievability coefficient. Through the command 'verify', the whole KB can be scanned to ensure that no KU refers to a nonexistent one. The commands 'load' and 'save' allow a verified KB to be loaded from or saved to disk. The knowledge base browser consists of a single scrollable list pane providing a view on the KB. It has a single menu.

Having selected a KU in the knowledge base browser, a *knowledge unit browser* can be

[4] Smalltalk-80 is a trademark of ParcPlace Systems. The current prototype of **IDIoT** was developed on both a Macintosh II and a Sun 3/60. The program uses 200K, and its associated knowledge base, 500K for about 300 knowledge units. Given the simplicity of the example texts (both in terms of syntax and of inferences), the latter number suggests that a knowledgê base for the interpretation of even 'children stories' would be several orders of magnitude larger than the current one.

opened. It allows the user to define the associations, candidates, expansion procedure, and ports of a KU, and indicate whether the KU is innate or not, a relay or not. The user can also specify the names and order of constraints. Automatic checks force the user to enter a syntactically correct expansion procedure. In particular, the system verifies that a user variable is bound to a cluster (through a getCluster or getNewCluster operation) before it is referred in another operation. For conditional instructions, the specified constraint must already exist. The system also verifies that all KUs referred to in the constraints of a KU have delays assigned to them. The knowledge unit browser consists of five top buttons, two bottom buttons, and a large scrollable list pane in the middle. The mutually exclusive top buttons drive both the menu and contents of the list pane. The two bottom buttons merely act as on/off toggles.

Selecting a constraint, the user can open a *constraint browser*, which allows the specification of the triggers, inputs, exceptions, and outputs of a constraint. The triggers can be marked as ordered or not. When defining an output, the user is prompted for a target KU and then for a signal to send.

Snapshots of the three browsers, displaying all menus, are included in figures 3.3 to 3.9. Menu entries should be self-explanatory and I will not elaborate on the use of these tools that operate much like the Smalltalk-80 browser (Goldberg, 1984).

3.5.1 The Knowledge Base Browser

See figure 3.3.

3.5.2 The Knowledge Unit Browser

See figures 3.4 to 3.8.

3.5.3 The Constraint Browser

See figure 3.9.

3.6 Details of IDIoT

In this section, I present the details of the processing of a text by **IDIoT**. First, I describe the task of 'knowledge' specification by the user through the browsers introduced in the

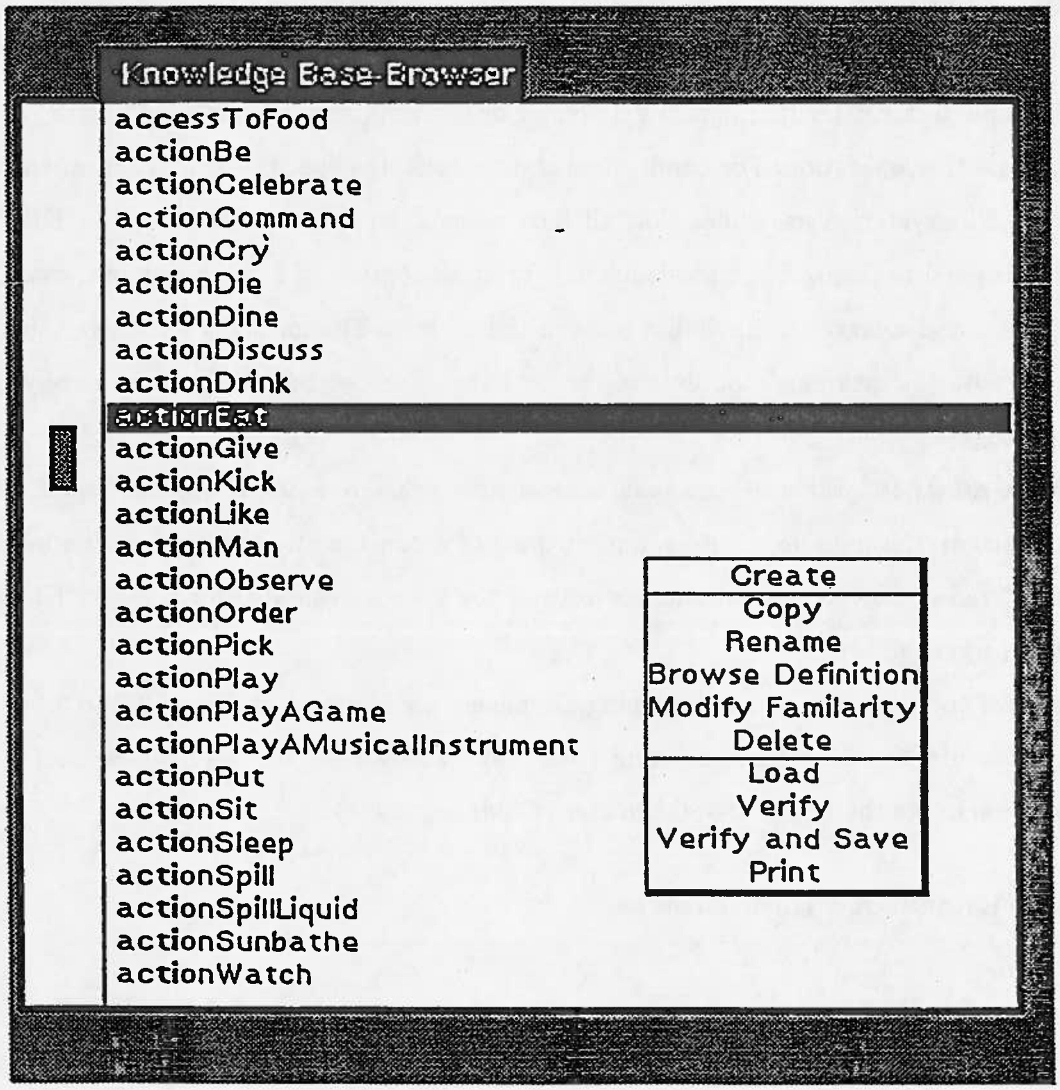
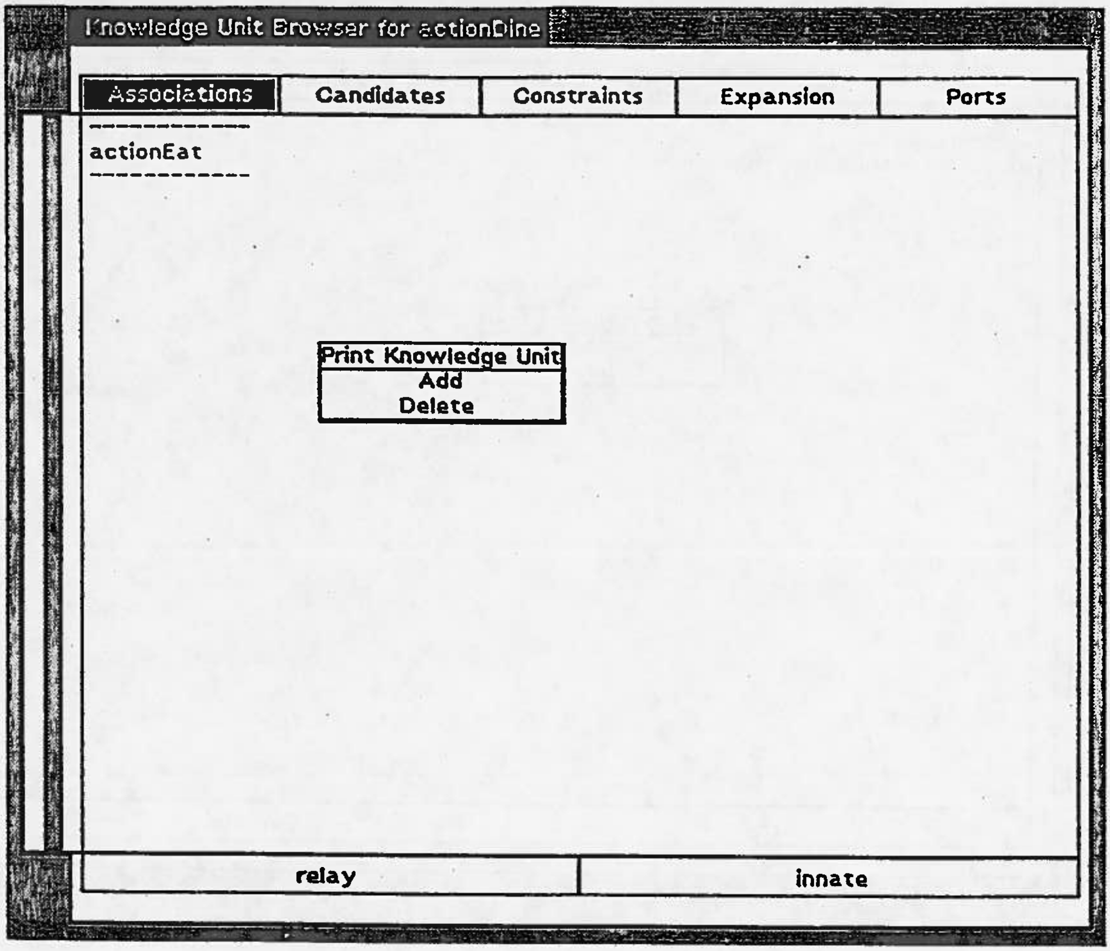


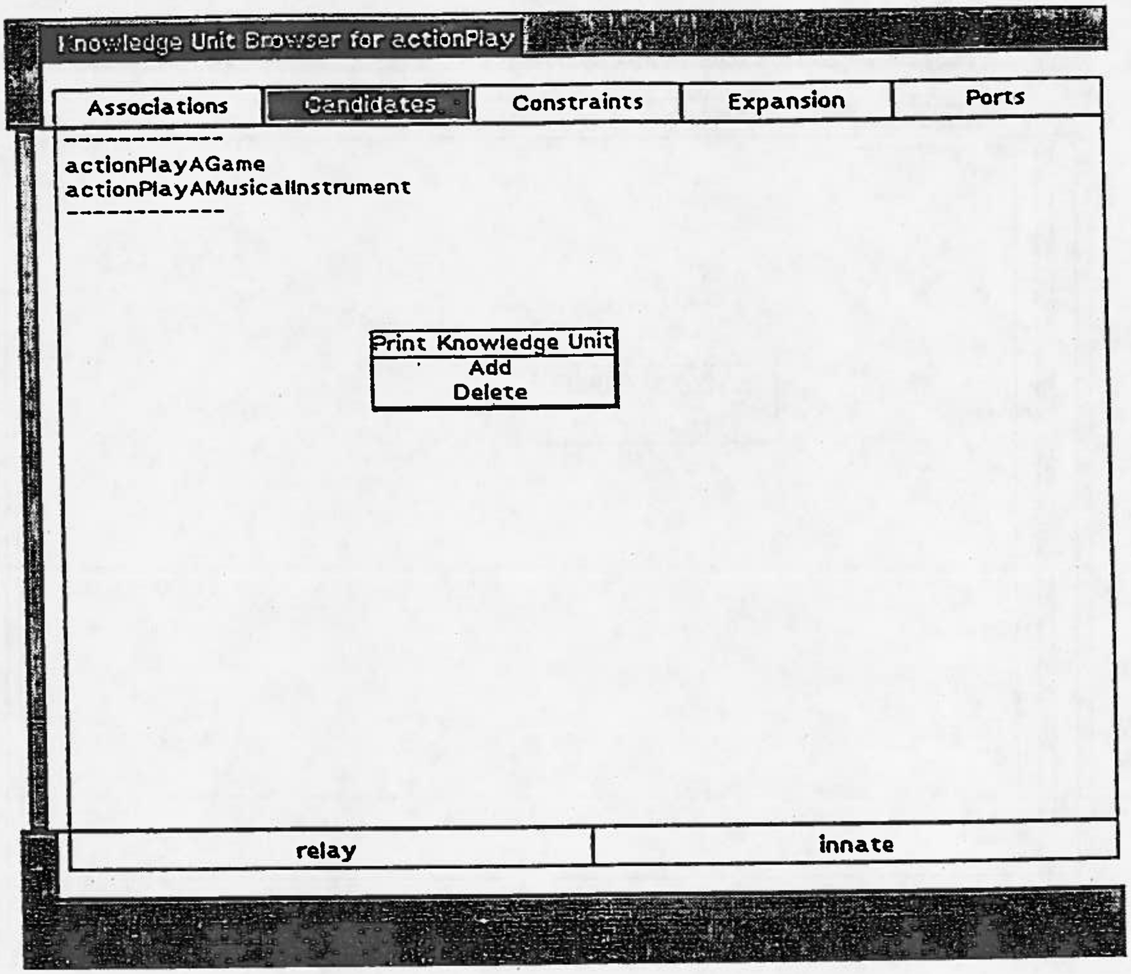
Figure 3.3: The Knowledge Browser



Knowledge Unit Browser for actionDine

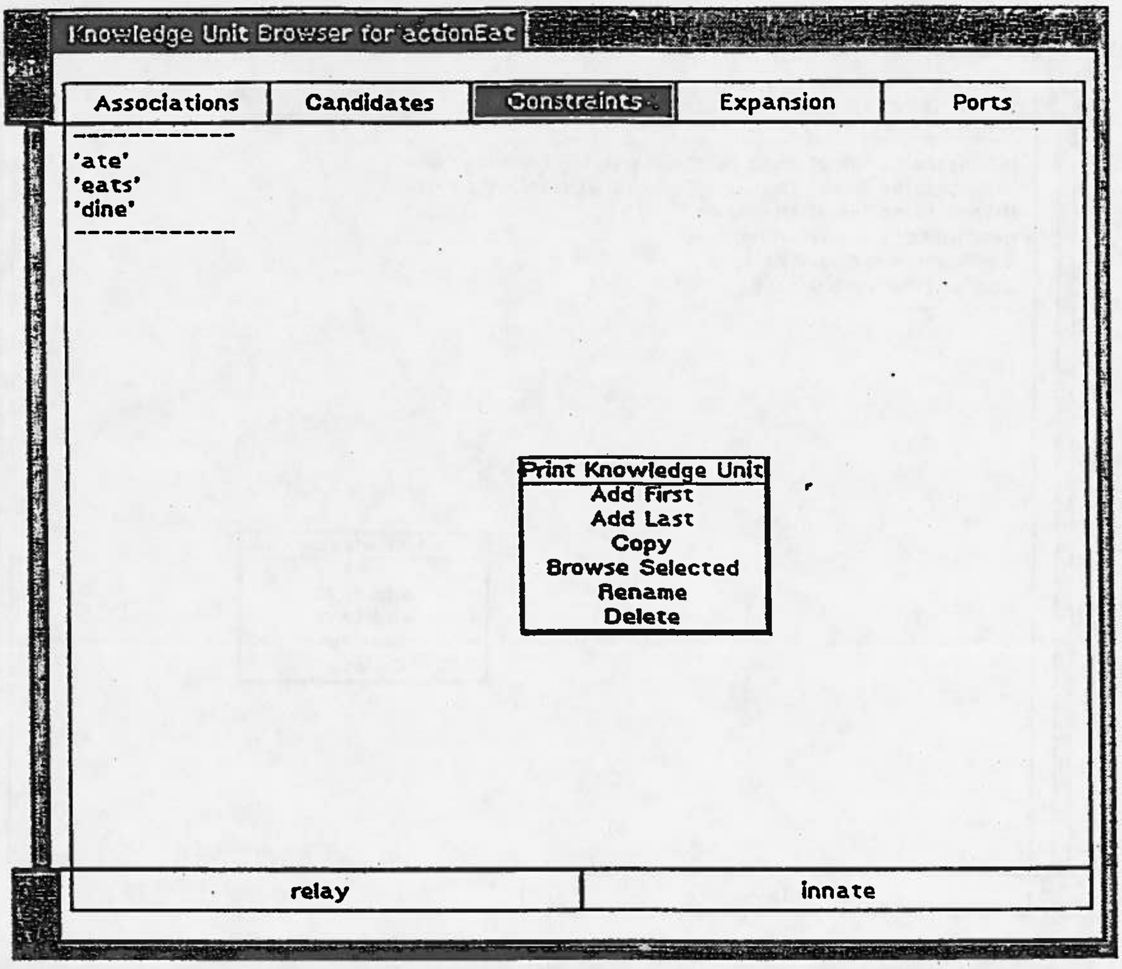
```
------------
actionEat
------------
```

Figure 3.4: The Knowledge Unit Browser: Associations



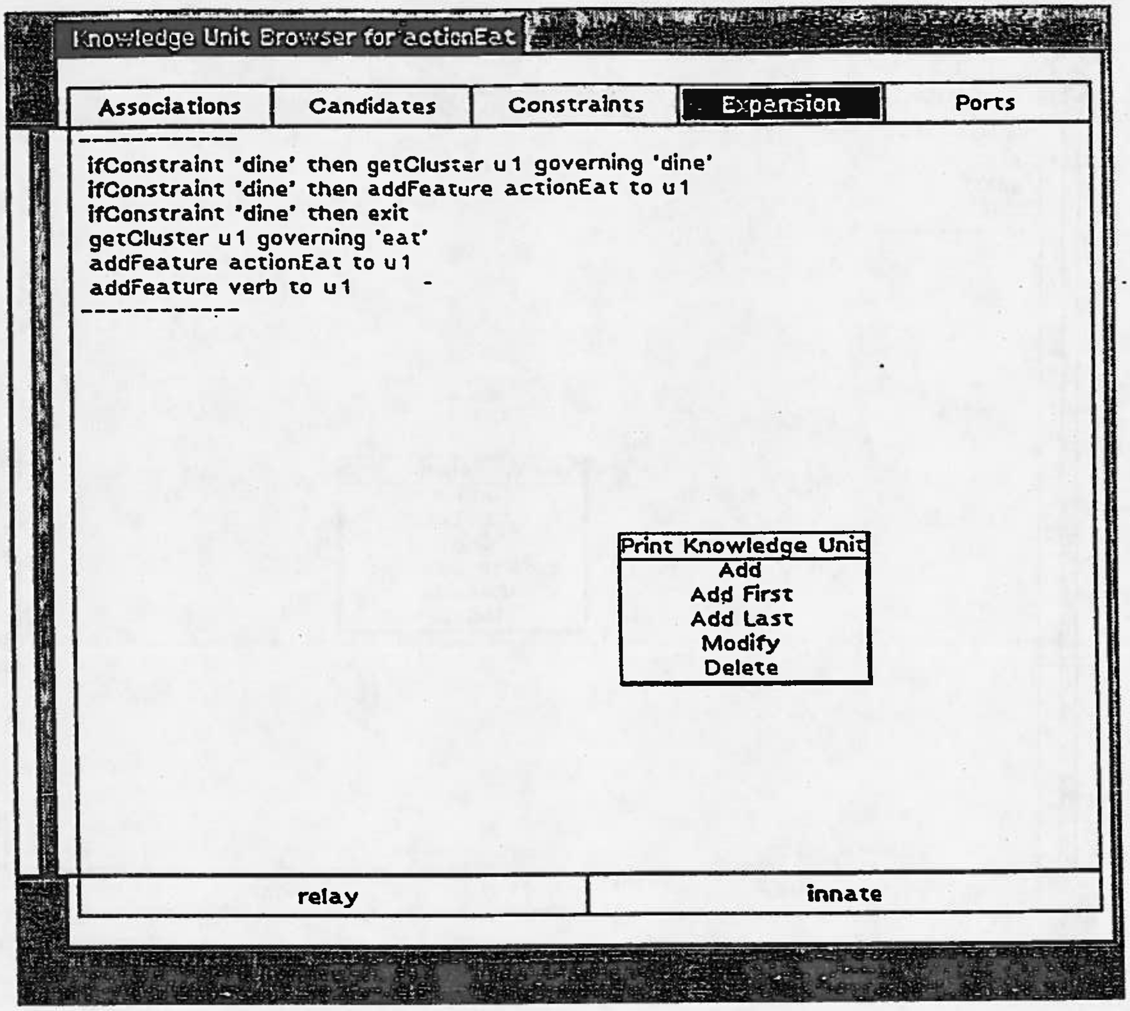
```
------------
actionPlayAGame
actionPlayAMusicalInstrument
------------
```

Figure 3.5: The Knowledge Unit Browser: Candidates



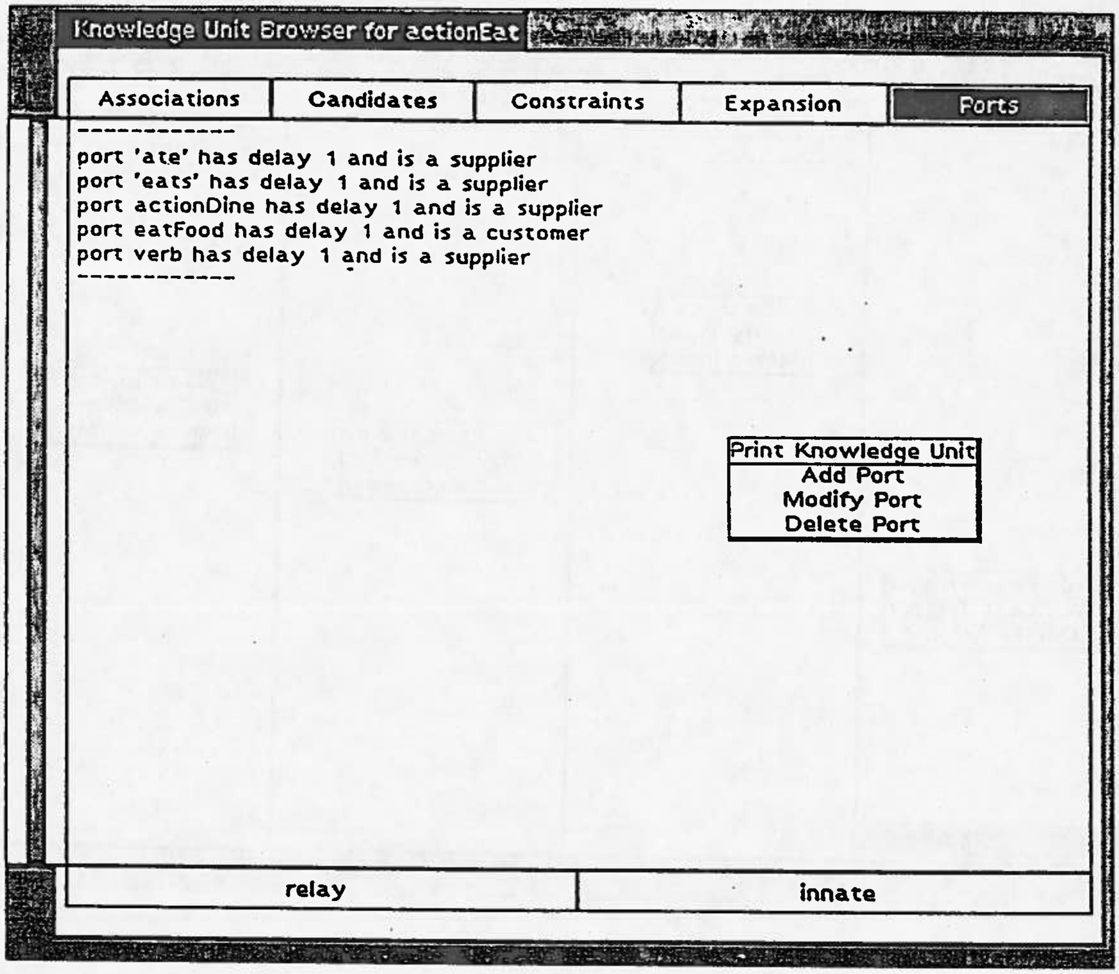
```
-------------
'ate'
'eats'
'dine'
-------------
```

Figure 3.6: The Knowledge Unit Browser: Constraints



```
------------
ifConstraint 'dine' then getCluster u1 governing 'dine'
ifConstraint 'dine' then addFeature actionEat to u1
ifConstraint 'dine' then exit
getCluster u1 governing 'eat'
addFeature actionEat to u1
addFeature verb to u1
------------
```

Figure 3.7: The Knowledge Unit Browser: Expansion Procedure



```
------------
port 'ate' has delay 1 and is a supplier
port 'eats' has delay 1 and is a supplier
port actionDine has delay 1 and is a supplier
port eatFood has delay 1 and is a customer
port verb has delay 1 and is a supplier
------------
```

Figure 3.8: The Knowledge Unit Browser: Ports

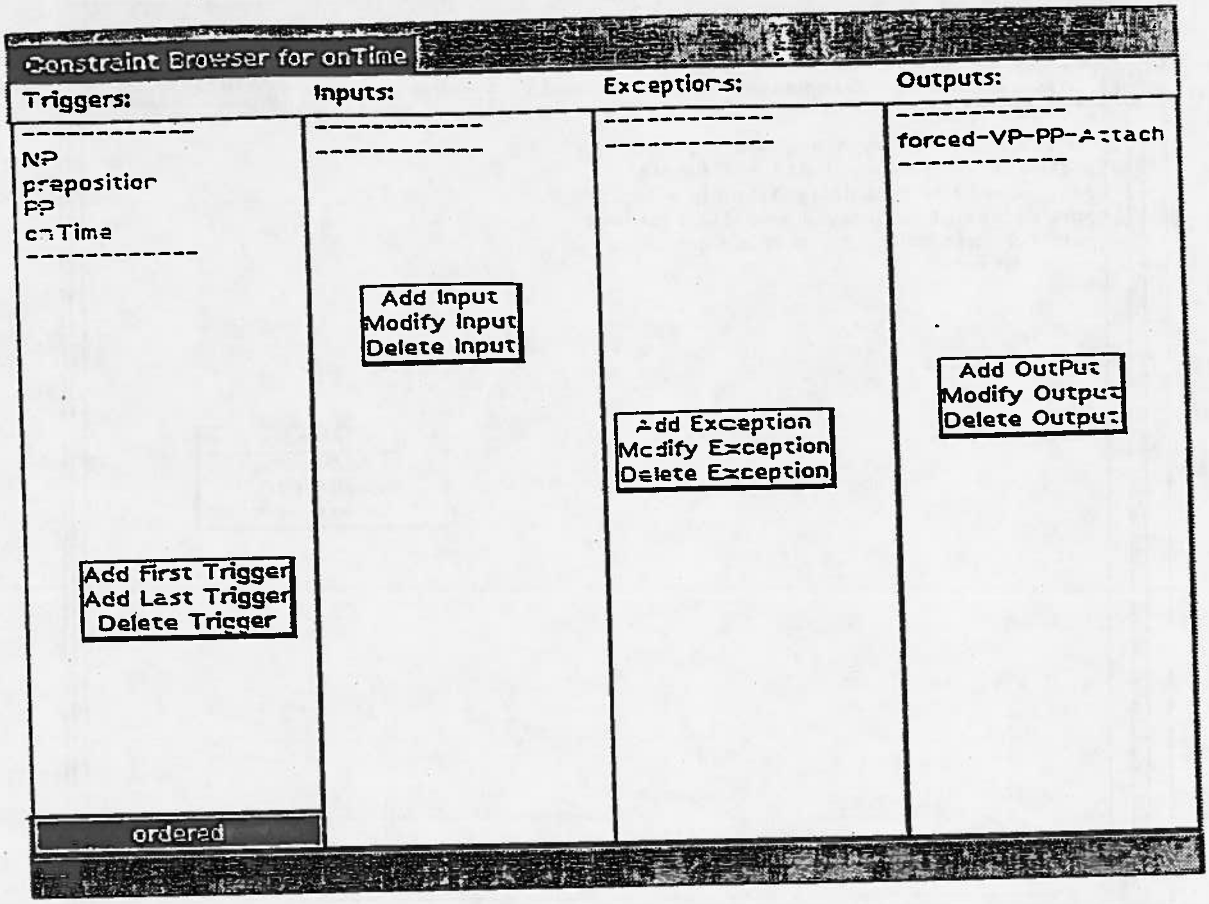


Figure 3.9: The Constraint Browser

previous section. Then I explain some of the network processes executed by the memory manager. Finally, I summarize the algorithms that control the behavior of an individual knowledge unit. I must emphasize that, as in the rest of this chapter, design decisions regarding the representational scheme (*e.g.,* necessity of triggers and exceptions, of an expectation signal) and its corresponding algorithms (*e.g.,* precedence of inhibition over forced detection, redundant states) do not necessarily proceed from psychological evidence, but rather from the engineering goal of getting a programmable comprehension tool that works.

3.6.1 'Knowledge' Specification

The user of **IDIoT** inputs both the structure and the behavior of individual KUs, which are stored in a knowledge base (KB). The user may specify several knowledge bases but must select a single one at the start of each interpretation. The selected KB constitutes the contents of the static memory of the system for the current interpretation. The ports of all KUs in the selected KB define the topology of the network of KUs forming static memory.

The information specified in a KU is strictly local to it, that is, independent of any other KU and of any global (network-wide) information. The knowledge base browser (figure 3.3) allows the user to create a new KU in a KB by giving it a unique name. A new or existing KU can have its definitions modified by opening a knowledge unit browser (figures 3.4 to 3.8) on it.

Once opened on a KU x, the knowledge unit browser allows the following actions:

- By selecting the 'Associations' button, the user can see the list of associations of x, KUs to which a forcedDetection signal is sent upon the detection of x. The user may add or delete associations to the list.

- By selecting the 'Candidates' button, the user can see the list of candidates of x, KUs to which a confirmation signal is sent upon the detection of x. The user may add or delete candidates to the list.

- By selecting the 'Constraints' button, the user can see the ordered list of the names of the constraints of x. A constraint browser (figure 3.9) must be opened to examine the contents of a specific constraint. The user may add a constraint to the beginning or the end of the list. Constraints may also be deleted.

- By selecting the 'Expansion' button, the user can see the ordered list of memory operations that form the expansion procedure of x. The user may add an instruction to the beginning or the end of the procedure, or after a selected instruction. An instruction is inserted in the procedure if and only if it has valid syntax (as defined in subsection 3.4.2). A conditional instruction must refer to an existing constraint of x, and all non-retrieval operations (*i.e.,* all instructions except for the getCluster series) must refer to user variables for which a retrieval operation existed in the procedure previously. (The first user variable of a 'find' instruction does not obey this rule because it is set to a retrieved cluster.) For example, the instruction

  addFeature actionEat to u1

  is valid only if

  getCluster u1 ...

  precedes it in the expansion procedure. This rule is also enforced when an instruction is deleted from an expansion procedure. Failure to comply with this rule results in the display of an appropriate warning to the user, as well as the rejection of the instruction.

- By selecting the 'Ports' button, the user can see the list of ports of x. The customer ports of a KU are those ports that are used to send out a signal to another KU; the supplier ports, to receive a signal from another KU. From the constraints of x, the system can infer which ports are suppliers (corresponding to the triggers, inputs, and exceptions of the constraints of x) and which are customers (corresponding to the output strategy of the constraints). Additional customer ports corresponding to KUs not referred to in the constraints of x may be specified: regardless of the satisfied constraint, a presence signal is sent to these ports upon the detection of x. Also, a KU may receive a signal from itself, in which case the port is considered to be a supplier port. Modifications to the constraints of x cause the system to recategorize all its ports. A delay must be specified for each port of a KU before the knowledge unit browser can be closed. Ports can be added or deleted, or have their delays modified.

- By selecting the 'innate' button, the user indicates that the KU corresponds to an innate feature, that is, to a feature that is automatically activated (*i.e.,* becomes detected) if its name is found as a string of characters (delimited by spaces or punctuation signs) in the input text. For example, the feature **'John'** is an innate feature that is automatically activated when the word 'John' is read from the input text.

An innate feature has no constraints, as it is directly activated by the input text. It may have associations, to which a presence signal, not a forcedDetection signal, is sent. At least one association must be specified before the knowledge unit browser of an innate feature can be closed. An innate feature may also have candidates, an expansion procedure, and ports.

- By selecting the 'relay' button, the user indicates that the KU is a relay, and thus must only consist of ports. Therefore the system only allows the specification of ports, which must be explicitly identified as either supplier or customer ones. An innate feature cannot be a relay.

The system automatically establishes whether a KU is buildable from the presence of 'test' or 'find' instructions in its expansion procedure. A buildable feature may have only triggers and outputs in its constraints. The system writes the phrase "is buildable" when a buildable feature is printed.

The knowledge base browser cannot be closed if any KU refers to an non-existent KU. In other words, the system enforces referential integrity throughout a knowledge base.

Finally, the constraint browser (figure 3.9) of a KU allows the user to specify the triggers, inputs, exceptions, and output strategy of a particular constraint. Triggers may be considered ordered or not through the use of the 'ordered' button. And to specify an output of a constraint, the user must give the target KU and the signal to send it. A constraint browser cannot be closed unless at least one trigger is specified.

3.6.2 Network Processing

The specification browsers operate within a ST-80 image (Goldberg, 1984) in which also resides the code that implements **IDIoT**. In order to start an interpretation, the user selects an input text (in an ASCII file) and a knowledge base. The default settings for the different system parameters (see section 3.7) of the time-constrained memory of **IDIoT** can then be altered. Once these parameters have been specified, a special event is sent to the memory manager to start the reading. The memory manager reads the input text one character at a time, each character 'costing' one character quantum of time. A word is considered to be a string of characters delimited by spaces or punctuation signs. Each time a word is found, the memory manager checks whether an innate feature of that name exists. If not, a warning stating that the word is unknown to the system is displayed, and the reading continues. Conversely, if the word is recognized, the memory manager sends a forcedDetection signal to its corresponding innate feature. This signal will cause the innate feature to send signals to its customers and request the execution of its expansion procedure by the memory manager. In turn, the signals sent out by the innate feature will eventually lead to the activation of other KUs, which, upon their detection, will send out signals to their customers and have their respective expansion procedures execute. The execution of an expansion procedure constructs or modifies clusters in dynamic memory. Once the text has been completely input, a browser opens on the interpretation generated for the text, that is, on the contents of the dynamic memory at the end of processing.

In order to handle sentences in a simple way, the current input process of the memory manager considers that a sequence consisting of a period followed by one or more spaces, followed by a capital, indicates the end of a sentence and the beginning of a new one. The current prototype requires that the user define the features **endOfSentence** and **startOfSentence** in the knowledge base in order for the system to recognize these features automatically. Features corresponding to the punctuation signs must also be specified by the user if they are to be processed automatically by the system rather than ignored.

Each exchange of signals in the system is routed through the memory manager, which computes the actual arrival time of a signal from:

- The delay associated with the port on which the signal is sent.

- The retrievability coefficient of the receiver.

- The nature of the signal.

A forcedDetection or forcedInhibition signal takes epsilon time to reach its destination. In other cases, the actual delay is computed as the product of the specified delay with the retrievability coefficient of the receiver. The memory manager keeps a schedule of all signals to be received and their time of arrival to their destination. This schedule is used to simulate real-time processing of communications in memory.

Independently of reading in the text, the memory manager has the ability to execute sequentially the expansion procedures of the KUs that become detected. Procedures are executed on a first-requested—first-executed basis. In order to execute an expansion procedure, the memory manager attempts the instructions of the procedure in order. Should an error occur (*e.g.*, failed getCluster operation, non-existent feature in a moveFeature, non-existent cluster in a removeCluster operation), the system will report to the user the simulated time of the error, the KU whose expansion procedure crashed, and the guilty instruction. It will then open a browser on the contents of the dynamic memory in order for the user to debug the procedure. The current prototype abandons further processing of the text in the event of such an error.

An expansion procedure has the ability to create new clusters (by means of the getNewCluster operation) and to access existing ones (using the getCluster operations). All data structures and algorithms for these tasks are specific to the implementation of **IDIoT**, and thus are taken to be of little interest to the reader. The rules controlling the membership of a cluster to one of the partitions of dynamic memory have been explained in section 3.2. These rules are enforced by the memory manager, which has the ability to directly modify the membership of a cluster to one of the partitions of dynamic memory. Both the capacity and membership constraints of the partitions of dynamic memory are reviewed only for getCluster and removeCluster operations. As previously mentioned, these operations, as well as the findInclusiveReference and findExactReference instructions, also involve traversing dynamic memory (going through clusters in a breadth-first manner), a process that consumes time (see subsection 3.4.2). All other operations of an expansion procedure are taken to execute in epsilon time.

The memory manager is also responsible for trying to execute the expansion procedure of a buildable feature (*ibid.*) that has a triggered constraint. Recall that a buildable feature is one whose expansion procedure comprises 'test' or 'find' instructions. Requests from a buildable feature to the memory manager asking to attempt to execute its expansion procedure are treated much like requests from non-buildable features to have their expansion procedure executed, but with the following differences:

- If the expansion procedure of a buildable feature succeeds, the memory manager sends the KU a forcedDetectionSignal and the feature does not request, upon its detection, the execution of its expansion procedure.

- If the expansion procedure of the buildable feature fails, then the manager registers the feature on a list of buildable features waiting to be built, marking down the time after which the feature should be purged from the list. In other words, the manager ensures that a buildable feature has a short amount of time to have its expansion procedure succeed. Failure to do so, and thus to become detected, intuitively corresponds to the situation in which the dynamic constraint that the buildable feature embodies fails.

- If the expansion procedure of a buildable feature fails, the memory manager has the ability to 'restore' the contents of dynamic memory to what they were before the unsuccessful attempt. In the current prototype, trying to execute the expansion procedure of a buildable feature creates a set of 'temporary links' between clusters. If the expansion procedure succeeds, then these temporary links graduate to the status of 'permanent' links; otherwise they almost instantly vanish because of decay. In other words, without going into implementation details, the current prototype uses a mechanism that does not involve backtracking when attempting to execute the expansion procedure of a buildable feature.

- Within the time allocated to a buildable feature to be successfully expanded, the memory manager will attempt to execute the expansion procedure of the buildable feature each time a new feature has its procedure executed, that is, each time the contents of the dynamic memory change.

Finally, the memory manager may receive a request to attempt to execute the expansion procedure of a buildable feature that has all its triggers but not a satisfied constraint because a submission signal was received for some of its triggers. In this case, the memory manager attempts the expansion procedure of the buildable feature only once and sends back a forcedDetection signal in case of success, or a forcedInhibition signal in case of failure. In other words, whereas a buildable feature with a satisfied constraint can have its expansion procedure attempted several times, a buildable feature that has received some submission signal for one or more of its triggers is given only one chance of succeeding.

3.6.3 Detailed Algorithms of a Knowledge Unit

The tasks of the memory manager represent only one of the two facets of the 'trivial' algorithm of time-constrained memory. The other facet is the signal processing algorithm that all knowledge units run asynchronously and in parallel. This algorithm is detailed below. It uses only information local to each KU and involves only simple operations on the signals each KU receives. A knowledge unit can be in one of several modes, each mode recognizing certain inputs and ignoring all others.

Each KU executes the following basic reception algorithm each time it receives one or more new signals:

1. Receive all new signals at time *t*. Once the memory manager has computed the actual time of arrival of a signal, it places the signal and its time of arrival in the queue of incoming signals of the receiving port. Each KU keeps a queue of pairs of the form $\langle incomingSignal, arrivalTime\rangle$ for each of its supplier ports. A KU receives all new signals at time *t* by retrieving all signals of its input port queues that have *t* as their arrival time. The signals received at *t* are called the current signals.
2. Independently of its current mode, the KU verifies that it has not exceeded the current deadline of its mode. If it has, it immediately reverts back to *idle mode.* In other words, at a given point in time, a KU is in a certain mode. All modes, except idle mode, last a fixed amount of time. Once the time allocated to a mode has been reached, the KU automatically reverts back to idle mode.
3. If the current signals include a forcedInhibition signal, then the KU immediately reverts back to idle mode and ignores all other current signals.
4. ElseIf the current signals include a forcedDetection signal, then the KU immediately goes to *detected mode.* Both forcedInhibition and forcedDetection signals are always routed by the memory manager to the manager's port of a KU.
5. Else, the KU goes to the algorithm of its current mode.

The algorithm for the *detected mode* is:

1. If the KU is a relay feature, it sends a presence signal to its its customers.
2. If the KU is an innate feature, it sends a presence signal to all its associations. Otherwise, it sends a forcedDetection signal to all its associations.
3. The KU sends a confirmation signal to all its candidates.
4. The KU sends a reinforcement signal to all KUs to which it sent initially a submission signal and that have sent back a confirmation signal. This step corresponds to the reinforcement of all missing features of the triggered constraint, if any.
5. The KU sends the specified signals to the customers listed in the outputs of the satisfied constraint, if any.

6. The KU sends a presence signal to customers to which it is not sending a reinforcement signal and that are not part of the outputs of the satisfied constraint.
7. The KU requests the execution of its expansion procedure by the memory manager and reverts back to idle mode.

The algorithm for the *idle mode* follows:

1. If the current signals include expectation signals, then follow the algorithm of the *expecting mode.*
2. ElseIf the current signals include submission signals, then execute the 'process submission input' method.
3. ElseIf the current signals include confirmation signals, then execute the 'start candidacy' method.
4. Else, execute the 'process all presence signals' method.
5. If the KU is a marker, then relay the current signals to the customers.
6. ElseIf a constraint is triggered (*i.e.,* has received a signal for one or more triggers), then execute the 'start satisfaction race' method.
7. Else, remain in idle mode.

All presence signals received by a KU start decaying upon reception. Each supplier port of a KU keeps a queue of presence signals individually associated with their time of reception. The 'value' of a supplier port for constraint satisfaction purposes is the sum of all its decayed presence signals at a particular point in time. Presence signals that decay below a certain threshold are automatically purged from their queue. The 'process all presence signals' method consists in computing the new value of all customer ports of a KU that receive new presence signals.

The 'process submission input' method follows:

1. Set submission mode deadline.
2. Process all presence signals.
3. If a constraint is fully triggered (*i.e.,* has a presence or submission signal for all its triggers), then:
   (a) If this is a buildable KU, then:
      i. Request an attempt to execute the expansion procedure.

```
        ii. If the attempt is successful, then:
            A. Send confirmation signal to submitters, that is, to the KUs from which
               a submission signal was received.
            B. Set mode to toBeReinforced.
        iii. Else (the attempt failed): Ignore submission signal and continue the idle
             mode algorithm.
    (b) Else (this is not a buildable):
        i. If the triggered constraint is satisfied, then:
            A. Send confirmation signal to submitters.
            B. Set mode to toBeReinforced.
        ii. Else (the triggered constraint is not satisfied):
            A. Send submission signal to the missing features of the triggered constraint.
            B. Set mode to toBeConfirmed.
4. Else (no fully triggered constraint): Stay in idle mode and continue the idle mode
   algorithm.
```

The 'start candidacy' method follows:

```
1. Set candidacy mode deadline.
2. Process all presence signals.
3. If there is no fully triggered constraint, then stay in idle mode.
4. Else (there is a fully triggered constraint), then:
    (a) If this is a buildable KU, then request attempt to execute the expansion procedure.
    (b) Else: Send a submission signal to customers.
5. Set mode to directImmediateCandidate.
```

The 'start satisfaction race' method follows:

```
1. Set satisfaction race deadline.
2. Process all presence signals.
3. If the triggered constraint has exceptions, then:
    (a) If this is a buildable KU, then:
        i. Request attempt to execute the expansion procedure.
        ii. Set mode to toBeBuiltImmediate.
    (b) Else (this is not a buildable KU):
```

        i. Send submission signal to missing features and to customers.
        ii. Set mode to endCandidate.

4. Else (the triggered constraint has no exceptions):
   (a) If buildable, then:
        i. Request attempt to execute the expansion procedure.
        ii. Set mode to toBeBuiltImmediate.
   (b) Else (not buildable):
        i. Send submission signal to missing features and to customers.
        ii. Set mode to immediateCandidate.

The algorithm when in *directImmediateCandidate mode* has the KU waiting for a confirmation signal from a customer. Upon reception of such a signal, the KU becomes detected. All other signals are ignored.

The algorithm when in *directEndCandidate mode* is the same as the preceding one except for the fact that input signals are processed only at the end of the time allocated to the race of this mode.

The algorithm when in *immediateCandidate mode* follows:

1. Process only confirmation signals.
2. If a constraint has received a confirmation signal from each of the suppliers it sent a submission signal to, then it becomes detected.
3. Else remain in this mode.

The algorithm when in *endCandidate mode* is the same as the preceding one except for the fact that input signals are processed only at the end of the time allocated to the race of this mode.

The algorithm when in *toBeReinforced mode* ignores all signals but reinforcements signals. Once the KU has received a reinforcement signal, it relays this signal to the KUs from which it received a confirmation signal and changes its mode to detected.

The algorithm when in *toBeConfirmed mode* follows:

1. Ignore all signals but confirmations signals.
2. For each confirmation signal:
   (a) If this is the last confirmation needed then:

    i. Send confirmation signal to submitters.
    ii. Set mode to toBeReinforced.

  (b) Else (this is not the last confirmation): Take it out of the list of KUs from which the KU still needs confirmations.

The algorithm when in *toBeBuiltImmediate mode* simple ignores all signals except for a forcedDetection signal from the memory manager.

When in *expecting mode*, a KU ignores all signals but presence signals for its triggers. The KU goes to detected mode as soon as it has received all the triggers for any of its constraints.

Finally, I remark that even though the user specifies an order among the constraints of a KU, the system tries to satisfy all constraints in parallel. Only then does the system refer to the user-specified order to decide which of the satisfied constraints is the one that causes the detection of the KU. Therefore, the number of constraints in a KU does not correlate to the complexity of the constraint satisfaction step. In other words, we avoid the problems that could result from the sequential processing of a large number of constraints.

3.7 An Annotated Example

To recapitulate the different facets of the proposed model of time-constrained memory, let us consider the processing of the sentence "John drinks gin". All linguistic examples follow a simple grammar derived from that of Winograd (1983). For the sake of simplicity, I will consider only two of the meanings of 'gin': the card game and the drink. Some of the KUs used in this example have been listed in subsection 3.4.3, and the structure of most others can be inferred from the exchange of messages. Also, I want to highlight only the most important aspects of the processing, and thus I have taken out several exchanges of messages not germane to the discussion at hand.[5]

Once the user of IDIoT has started the system and given an input text, he or she can modify the defaults of the different parameters of time-constrained memory. Here are the default settings for the examples of this dissertation:

- Candidacy length = 200

[5] The actual trace is verbose, listing every message exchange and state change (see appendix A). For all the running examples of the current prototype, traces total more than 10 megabytes of text.

- Expectation length = 100
- Submission length = 50
- Decay factor = 0
- Epsilon quantum = 0.01
- Character quantum = 10
- Detection threshold = 1
- STM capacity = 50
- Working memory capacity = 7

I emphasize that these values are arbitrary and do not proceed from psychological evidence. They have been chosen to simplify the processing as much as possible by having a large STM, long races, and no decay.

The values for the different special signals are the following:

- forced detection signal = −1
- confirmation signal = −2
- expectation signal = −3
- inhibition signal = −4
- reinforcement signal = −5
- submission signal = −6

Again, for the sake of simplicity, the presence signals used by the KUs of this example have systematically been given their maximum value of 1.

The system is designed to automatically insert a special innate feature called **startOfSentence** at the beginning of the text and after each period. Another special innate feature, **endOfText**, is inserted at the end of the text. The processing of the text thus starts, at time 0, with the recognition of **startOfSentence** which sends a presence signal to its customers. Then, at time 3, the word 'John' is recognized. Here is its simple definition:

```
innate KU 'John':
 associations: johnPerson
 expansion:
      getNewCluster u1
      addFeature singular to u1
      addFeature proper to u1
      addFeature person to u1
      addFeature male to u1
      addFeature 'John' to u1
 ports:
      ...
```

The features **singular**, **proper**, **person**, and **male** are defined elsewhere in the KB. For example, **proper** is a relay with all known proper nouns as suppliers and all rules relevant to proper nouns as customers.

All features referred to in the expansion of 'John' are sent a forced detection signal:

```
innate KU 'John' read in at 3
  is recognized
  manager expands
  manager sends forcedDetectionSignal from manager to singular  arriving at 3.01
  manager sends forcedDetectionSignal from manager to proper  arriving at 3.01
  manager sends forcedDetectionSignal from manager to person  arriving at 3.01
  manager sends forcedDetectionSignal from manager to male  arriving at 3.01
```

Throughout the trace, 'manager' refers to the memory manager and the phrase 'manager expands' means that the memory manager executes the expansion procedure of the corresponding KU.

The feature **person** will in turn lead to the detection, at 3.02, of its two associations **rational** and **animate**. At 3.01, feature **proper** becomes detected and sends a presence signal to its customers, one of which is **NP**. Given an *a priori* delay of 1 between these two features (as with the vast majority of KUs in this example), at 4.01, **NP** receives this signal, which triggers and satisfies its constraint called *proper*. Among the many customers of **NP** that are sent a present signal is the feature **s-Start**. This feature was triggered by **startOfSentence**. Once the presence signal from **NP** is received, **s-Start** becomes detected, capturing the fact that the sentence starts with an NP (as opposed, for example, to a verb that could indicate the imperative mood). Through the expansion of **s-Start**, the cluster associated with the NP has features **topNP** and **subject** added to it. (The feature **topNP** keeps track of the most referable NP of a sentence.) The feature subject captures the intuition that the first NP of a non-imperative sentence will be assumed to be the subject unless subsequently proven otherwise. At this point in time (5.01), dynamic memory contains a cluster for the NP with **NPhead** governing the cluster associated with John.

Each letter of the word 'John' takes a character quantum, in this case, 10 time units, to read. Hence the next word is input 40 time units after 'John', at time 43. The innate feature **'drinks'** sends a presence signal to the feature **actionDrink**, whose definition follows:

```
KU actionDrink:
 constraint 'drinks':
  triggers: 'drinks'
  inputs:
      verb has a weight of 1
 expansion:
      getCluster u1 governing 'drink'
      addFeature actionDrink to u1
      addFeature verb to u1
 ports:
      'drinks' has a delay of 1 and is a supplier
      drinkBeverage has a delay of 1 and is a customer
      verb has a delay of 1 and is a supplier
```

Feature **actionDrink** receives the presence signal from **'drinks'** at 44 and becomes a candidate. It sends a submission signal to its missing feature **verb**. Feature **verb** simply relays the submission signal to its customer **VP**. This is used for encapsulation, that is, in order to avoid having every known verb of the system have **VP** as a customer. This constitutes a typical role for relays.

VP relays the submission signal received from verb to its numerous customers including feature **subj-verb-rel** which captures the subject-verb relationship. The latter feature has already been triggered by the feature **subject** and, once it receives the submission signal from its input VP, has one of its constraints satisfied. At that point, **subj-verb-rel** does not become activated, but rather sends a confirmation signal back to **VP**. This confirmation signal is relayed from **VP** to **verb** to **actionDrink**. Since the triggered constraint of **actionDrink** does not have exceptions, **actionDrink** can become detected as soon as it receives a confirmation signal from all features to which it sent a submission signal, that is, from **verb**. At this point, **actionDrink** checks the deadline of its candidacy and forgets it since it has not been exceeded, becomes detected, and sends a reinforcement signal to **verb** and a presence signal to its customer **drinkBeverage**. The features **verb**, **VP**, and **subj-verb-rel** eventually become activated:

```
KU actionDrink in mode immediateCandidate works at 50
  receives confirmationSignal on verb
  confirmers are verb
  manager sends reinforcementSignal from actionDrink to verb arriving at 51
  becomes detected at: 50
  forgets end of race deadline
  manager expands
  manager sends presenceSignal from actionDrink to drinkBeverage arriving at 51
KU drinkBeverage in mode idle works at 51
  receives presenceSignal on actionDrink
KU verb in mode toBeReinforced works at 51
  receives reinforcementSignal on actionDrink
  manager sends reinforcementSignal from verb to VP arriving at 52
  becomes detected at: 51
KU VP in mode toBeReinforced works at 52
  receives reinforcementSignal on verb
  manager sends reinforcementSignal from VP to subj-verb-rel arriving at 53
  becomes detected at: 52
  manager expands
  manager sends forcedDetectionSignal from manager to activeMood  arriving at 52.01
  manager sends forcedDetectionSignal from manager to mainVerb  arriving at 52.01
  manager sends forcedDetectionSignal from manager to VPmods  arriving at 52.01
       ...
KU subj-verb-rel in mode toBeReinforced works at 53
  receives forcedDetectionSignal on VP
  becomes detected at: 53
  manager expands
  manager sends forcedDetectionSignal from manager to clause  arriving at 53.01
  manager sends presenceSignal from subj-verb-rel to directObject arriving at 54
  manager sends presenceSignal from subj-verb-rel to newFact arriving at 54
  manager sends presenceSignal from subj-verb-rel to sva-1 arriving at 54
  manager sends presenceSignal from subj-verb-rel to sva-2 arriving at 54
```

As a result of the expansion of **subj-verb-rel**, dynamic memory contains a cluster having the single feature **clause**. This feature governs a subcluster corresponding to the VP and another subcluster corresponding to the subject. The two customers **sva-1** and **sva-2** of **subj-verb-rel** are buildable features used to verify that the subject and the verb agree. They are discussed in chapter 5. As for feature **directObject**, it has its constraint triggered by the presence signal received from **subj-verb-rel**.

At time 103, the word 'gin' is input and sends a presence signal to **gin**. Feature **gin** sends a submission signal to **noun**, which relays it to **NP**, which sends a submission signal to **directObject**. The definition of **directObject** follows:

```
KU directObject:
% This feature takes the most reachable NP and makes
% it the directObject of the verb if the latter is not
% intransitive. It also makes this NP the top NP of the clause.
% The next line is automatically generated by the system
% to indicate the KU is buildable.
is buildable
 constraint c1:
  ordered triggers: subj-verb-rel NP
 expansion:
      getCluster u1 governing VP
      getCluster u2 governing topNP in u1
      removeFeature topNP in u2
      getCluster u2 governedBy mainVerb in u1
      testAbsenceOf intransitive in u2
      addFeature verbComplemented to u2
      getCluster u2 governing NP
      addFeature directObject to u1
      addSubCluster u2 to directObject in u1
      addFeature topNP to u2
 ports:
      ...
```

Feature **directObject** is buildable and therefore, upon receiving the submission signal from **NP** and having one of its constraints satisfied, requests that the memory manager check whether **directObject**'s expansion procedure succeeds. This procedure succeeds if the verb admits of a direct object. In the example, the expansion of **directObject** succeeds and **directObject** sends a confirmation signal back to **NP**:

```
KU directObject in mode idle works at 107
  receives submissionSignal on NP
  manager Port is 0
  in submission, a triggered constraint is c1
  satisfaction of c1 against threshold 1
  manager attempts dynamic construction for submission at 107
  manager sends confirmationSignal from directObject to NP arriving at 108.26
```

The confirmation signal will ripple from **NP** to **noun** to **gin**, which becomes detected. Feature **gin** has two candidates, **cardGameGin** and **alcoholicBeverageGin** to which it sends a confirmation signal in order to have their candidacies start. Each candidate sends a submission signal to its customers. The submission path **alcoholicBeverageGin**, **alcoholicBeverage**, **beverage**, **drinkBeverage** will eventually be built and lead to the confirmation of alcoholicBeverageGin, whereas a path from cardGameGin will reach a dead end with feature **actionPlayAGame**. Further explanations are provided in part 2 of this thesis. At the same time that these candidacies occur, the features **noun**, and then **NP**, will be reinforced and become activated. Feature **NP** will reinforce **directObject**, which also becomes detected. The following segment of the trace captures this sequence of events:

```
KU gin in mode immediateCandidate works at 110.26
  receives confirmationSignal on noun
  manager Port is 0
  confirmers are noun
  manager sends reinforcementSignal from gin to noun arriving at 111.26
  becomes detected at: 110.26
  forgets end of race deadline
  manager expands
  manager sends confirmationSignal from gin to cardGameGin arriving at 111.26
  manager sends confirmationSignal from gin to alcoholicBeverageGin arriving at 111.26
KU alcoholicBeverageGin in mode idle works at 111.26
  receives confirmationSignal on gin
  manager Port is 0
  start direct candidacy race ending at 311.26 for constraint gin
  manager sends submissionSignal from alcoholicBeverageGin to alcoholicBeverage arriving at 112.26
KU cardGameGin in mode idle works at 111.26
  receives confirmationSignal on gin
  manager Port is 0
  start direct candidacy race ending at 311.26 for constraint gin
  manager sends submissionSignal from cardGameGin to cardGame arriving at 112.26
  manager sends submissionSignal from cardGameGin to rulesOfGin arriving at 112.26
KU noun in mode toBeReinforced works at 111.26
  receives reinforcementSignal on gin
  manager sends reinforcementSignal from noun to NP arriving at 112.26
  becomes detected at: 111.26
KU alcoholicBeverage in mode idle works at 112.26
  receives submissionSignal on alcoholicBeverageGin
  manager Port is 0
  in submission, a triggered constraint is alcoholicBeverageGin
  satisfaction of alcoholicBeverageGin against threshold 1
  manager sends submissionSignal from alcoholicBeverage to aboutAlcoholBeverages arriving at 113.26
  manager sends submissionSignal from alcoholicBeverage to beverage arriving at 113.26
KU NP in mode toBeReinforced works at 112.26
  receives reinforcementSignal on noun
  manager sends reinforcementSignal from NP to directObject arriving at 113.26
  becomes detected at: 112.26
  manager expands
  manager sends forcedDetectionSignal from manager to NPhead  arriving at 112.27
  manager sends forcedDetectionSignal from manager to NPquant  arriving at 112.27
  manager sends forcedDetectionSignal from manager to NPmods  arriving at 112.27
  manager sends forcedDetectionSignal from manager to 3rdPerson  arriving at 112.27
  manager tries dynamic construction of sva-1 at 112.26
  manager tries dynamic construction of sva-2 at 112.26
      ...
KU cardGame in mode idle works at 112.26
  receives submissionSignal on cardGameGin
  manager Port is 0
  in submission, a triggered constraint is cardGameGin
  satisfaction of cardGameGin against threshold 1
  manager sends submissionSignal from cardGame to game arriving at 113.26
  manager sends submissionSignal from cardGame to aboutPlayingCards arriving at 113.26
KU rulesOfGin in mode idle works at 112.26
  receives submissionSignal on cardGameGin
  manager Port is 0
  in submission, a triggered constraint is cardGameGin
  satisfaction of cardGameGin against threshold 1
KU game in mode idle works at 113.26
  receives submissionSignal on cardGame
  manager Port is 0
  in submission, a triggered constraint is cardGame
  satisfaction of cardGame against threshold 1
  manager sends submissionSignal from game to actionPlayAGame arriving at 114.26
KU beverage in mode idle works at 113.26
  receives submissionSignal on alcoholicBeverage
  manager Port is 0
  in submission, a triggered constraint is alcoholicBeverage
  satisfaction of alcoholicBeverage against threshold 1
  manager sends submissionSignal from beverage to drinkBeverage arriving at 114.26
  manager sends submissionSignal from beverage to liquid arriving at 114.26
KU aboutAlcoholBeverages in mode idle works at 113.26
  receives submissionSignal on alcoholicBeverage
  manager Port is 0
  in submission, a triggered constraint is c1
  satisfaction of c1 against threshold 1
KU aboutPlayingCards in mode idle works at 113.26
  receives submissionSignal on cardGame
  manager Port is 0
  in submission, a triggered constraint is c1
  satisfaction of c1 against threshold 1
KU directObject in mode toBeReinforced works at 113.26
  receives reinforcementSignal on NP
  becomes detected at: 113.26
  manager expands
  manager sends forcedDetectionSignal from manager to verbComplemented  arriving at 113.27
  manager sends forcedDetectionSignal from manager to topNP  arriving at 113.27
  manager sends presenceSignal from directObject to coi arriving at 114.26
  manager sends presenceSignal from directObject to newFact arriving at 114.26
KU drinkBeverage in mode idle works at 114.26
  receives submissionSignal on beverage
  manager Port is 0
  in submission, a triggered constraint is c1
  satisfaction of c1 against threshold 1
  manager attempts dynamic construction for submission at 114.26
  manager sends confirmationSignal from drinkBeverage to beverage arriving at 115.52
KU actionPlayAGame in mode idle works at 114.26
  receives submissionSignal on game
  manager Port is 0
```

Eventually, the path of features that led to the detection of alcoholicBeverageGin will also become detected and the period of the sentence combined with the innate feature **endOfText**, will complete the processing. The resulting structure in dynamic memory is as shown below:

```
KU newFact that governs:
   cluster 12565 in working memory with features:
   KU VP
   KU mainVerb that governs:
     cluster 14282 in working memory with features:
     KU 'drink'
     KU 3rdPersonSingForm
     KU actionDrink
   KU verb
   KU activeMood
   KU verbComplemented
  KU VPmods
  KU subject that governs:
   cluster 10317 in working memory with features:
   KU NP
   KU NPhead that governs:
    cluster 257 in working memory with features:
    KU singular
    KU ‘John’
    KU proper
    KU person
    KU male
    KU rational
    KU animate
    KU 3rdPerson
   KU NPquant
   KU NPmods
  KU directObject that governs:
   cluster 10186 in working memory with features:
   KU NP
   KU NPhead that governs:
    cluster 7652 in working memory with features:
    KU ‘gin’
    KU singular
    KU gin
    KU noun
    KU 3rdPerson
    KU alcoholicBeverageGin
    KU tasteOfGin
    KU alcoholicBeverage
    KU aboutAlcoholBeverages
    KU beverage
    KU liquid
    KU aboutLiquids
   KU NPquant
   KU NPmods
```

3.8 Observations from Experiments

To conclude this chapter, I wish to summarize the different programmable facets of the proposed model of time-constrained memory and, with respect to their usage, briefly discuss some of the insights gained from experiments with the current prototype.

Recall that in the current prototype of **IDIoT**, memory is partitioned into a static and a dynamic component. Static memory acts as the 'knowledge base' (KB), whereas dynamic memory serves as the repository for the clusters constructed during understanding: the user specifies KUs, sets up the parameters of the memory model, and then starts an interpretation for a selected text. In accordance with a reader-based comprehension strategy, not only different sets of KUs (*i.e.*, different KBs), but also different settings of the parameters may lead to different interpretations.

Let us first investigate this assertion with respect to 'knowledge' specification. Static memory proceeds from the conceptualization of memory as a massively parallel network of simple computing units, the KUs. The connectionist characterization of this network as a graph with weighted links and with nodes that sum up their inputs to compare them to an activation threshold is enhanced in three ways:

1. In order to increase the programmability of the model (subsection 3.4.1):
    - A KU is permitted to have several constraints associated with it. These constraints are to be organized by the user in strict decreasing order of precedence so that only one may be satisfied at once.
    - The intuitive notions of triggers, inputs, and exceptions are introduced for constraint specification. For flexibility, triggers may or may not be ordered.
    - The intuitive notions of associations and candidates are introduced to identify specific relationships between a KU and some of its customers.
    - Weights are not associated with the links of the network, but rather with the inputs and exceptions of constraints.
    - Communication is asynchronous and an *a priori* communication delay is associated with each link, more precisely with each port sending a signal to another KU.
2. In order to model the generative ability of memory (*i.e.*, the construction of clusters), KUs may possess expansion procedures (subsection 3.4.2).
3. In order to model the assumption that KUs are ordered with respect to the intuitive notion of familiarity, each KU is assigned a retrievability coefficient (section 3.3).

Given this model of static memory, the specification of the KB is entirely left to the user:

- **Organizing KUs on the retrievability axis:** Recall that the actual delay associated with an exchange of a signal is computed as the product of the *a priori* delay with the retrievability coefficient of the receiver. The coefficient is not an integer but a floating point number. Because the current prototype does not support the dynamic modification of a KU's coefficient by another KU, the user can set the retrievability of KUs by following a simple heuristic such as assigning a coefficient of 1 to 'independent concepts' and only using this coefficient to define an ordering among 'related concepts'. For example, the words 'pen' and 'plane' are likely to be equally retrievable, but the interpretation of 'pen' as a male swan may be far less familiar than other 'meanings' of the word, granting it a significantly higher retrievability coefficient (*e.g.,* a value of 2 to make it twice as long to retrieve as more 'common' senses). Experiments suggest that the difference between the values of two coefficients must be several times (*i.e.,* at least an order of magnitude) larger than the epsilon quantum (see section 3.3) in order to become significant with respect to processing. Also, depending on the time-span of candidacies, too high a retrievability coefficient may make a KU systematically unreachable, and thus irrelevant for the interpretative process.

  From a quantitative viewpoint, tracking the 'relatedness' of two KUs is a difficult task. At a superficial level, it simply consists in checking whether or not their respective transitive closures of reachable suppliers and customers KUs intersect. But this is not sufficient; for example, KUs that access the same cluster may also be considered as 'related' (*e.g.,* the KUs for recognizing the direct and indirect object of a bitransitive verb, see chapter 5). A fundamental goal of the familiarity axis is to minimize, through ordering, the risks of conflicting concurrent detections of 'related' KUs. This is particularly important for a problem such as word sense disambiguation (see chapter 7). Other strategies to avoid such conflicts include refining or reordering constraints, using inhibition signals to implement a winner-take-all strategy, and using exceptions, as discussed in part 2 of the dissertation. The point is that ordering KUs along the familiarity axis constitutes a simple approach to reducing the risk of erroneous interactions resulting from a large number of simultaneous 'related' detections. The ordering also affects the interpretative process by indirectly creating an order of reachability (*e.g.,* for syntactic and semantic preferences) on which the construction of clusters is based.

- **Using associations**: Experiments suggest that associations are best limited to archetypical generalizations along the conceptual 'is-a' hierarchy. For example, the fact that a cat is a mammal suggests that the KU **cat** have the KU **mammal** as an association. A presence signal of varying strength (see below) should be used otherwise to capture the 'strength' of the generalization.

  Recall that associations are sent forcedDetection signals, and exceptions, forcedInhibition signals. These priority signals take epsilon time to reach their destination. An inherent danger in the abuse of associations and exceptions is the creation of a communication network where all timing aspects are in essence bypassed because of an excessive dependence on priority signals at the detriment of the user-specified communication delays and retrievability coefficients.

- **Using candidates**: Experiments suggest that the use of candidates is best limited to sets of mutually exclusive KUs. They are particularly appropriate for word sense disambiguation (chapter 7).

- **Using buildable features**: Buildable features do not follow the forward or backward chaining strategies proposed for constraint satisfaction (subsection 3.4.1), and thus do not lead to a sequence of detections (through paths of reinforcement signals). Therefore, their use should be limited to features whose detection relies on the contents of dynamic memory (*e.g.*, reference resolution features, see chapter 6). One difficulty with using a buildable feature is that the time of its detection is less predictable than that other features because it depends on the dynamic (*i.e.*, unpredictable) context.

- **Varying the strength of a presence signal output**: In order to notify a customer of its presence, a KU typically sends it a presence signal varying from 0 to 1, corresponding to the 'strength' of the sender's presence with respect to the receiver. For example, along the conceptual 'is-a' hierarchy, it is possible that a child would 'strongly' classify (*e.g.*, send an output of 0.8 or more) a dolphin as a mammal and 'weakly' classify (*e.g.*, send an output of 0.3 or less) it as a fish. Experiments suggest that the user avoid 'in-between' presence signals (*e.g.*, between 0.3 and 0.8) so that an accumulation of relatively weak signals does not lead too quickly to the satisfaction of a constraint. Choosing the strength of presence signals, the weights to use in constraints, and the delay to assign to exchanges constitute the trickiest tasks for the user of **IDIoT**, tasks I have found quite akin to the 'tweaking' of weights in local connectionist networks. However, only the setting of communication delays has been found to have an immediate and significant effect on the processing of the current examples.

Variations in the internal structure of KUs and in the delays of the communication network may modify the likelihood of detection of a KU, and thus the sequence of cluster operations, ultimately resulting in potentially different interpretations. Another source of divergence during the interpretative process of **IDIoT** is the specificity of KUs, which affects the length of paths of reinforcement signals. Recall that the longer the path, the more risk there is of having some part of it not have sufficient time to become reinforced and detected (see subsection 3.4.1). The specificity of human knowledge is still a controversial issue but it appears that we acquire both extremely specific rules (*e.g.,* the parsing rules of a child) as well as some more 'abstract' rules (*e.g.,* generalizations, schemas).

The scope of action of the user of **IDIoT** is not limited to 'knowledge' specification but also involves the setting of the parameters of the memory model. First, let us consider the user-specified parameters pertaining to the time-span of the different states of a KU. The values of these parameters directly constrain the chances of activation of KUs, and thus the interpretative process itself:

- **Candidacy Length**: This delay applies to constraint satisfaction, that is, defines the amount of time a KU has to satisfy its triggered constraint and become activated. Experiments suggest that a 'short' candidacy length typically reduces the number of detections dramatically, and thus the extent of the final interpretation. Conversely, a long candidacy length has little effect, specially if there is some decay, in that there is an upper bound on the time required for a chain of clusters to become activated. Because both word recognition (*i.e.,* recognition of innate features) and exchange of signals consume time, the length of a candidacy should be several orders of magnitude larger than the epsilon quantum in order to take into account the next word(s) of the input.

- **Expectation Length**: This delay defines the amount of time a KU has to receive one of its triggers and become activated. The setting of this parameter depends entirely on the role that the user is willing to grant to expectations. An excessively short delay (*i.e.,* close to the epsilon quantum) will not allow any expectations. An average delay (*e.g.,* a few orders of magnitude larger than the epsilon quantum) will allow consideration of further input and some immediate expectations (*e.g.,* semantic priming effects, see chapter 7). A longer delay will allow for more complex expectations (*e.g.,* involving inferences). An extremely long delay may lead to the detection of somewhat 'far-fetched' conceptual links.

- **Submission Length:** This delay defines the amount of time a KU has to receive all appropriate confirmation signals from a set of suppliers and send a confirmation signal to the KU that initially sent it a submission signal. Experiments suggest setting this delay, which defines the time-span of backward-chaining, to the same order of magnitude as the candidacy length, which defines the time-span of forward chaining. Because of the larger number of exchanges of signals required by backward-chaining, I typically have set the length of a submission to twice the candidacy length. Variations have basically the same effect as with the candidacy length. In both cases however, arbitrarily long delays seem psychologically implausible.

Second, despite the fact that, during the processing of a text, the user has no direct control on where a cluster resides in dynamic memory, the construction of clusters, that is, the interpretative process, is constrained by the values of the user-specified parameters pertaining to memory management:

- **Epsilon Quantum:** All memory parameters pertaining to time are based on the epsilon quantum that, in essence, defines the unit of granularity of time for memory. Further investigation of psychological and neuronal evidence is required before establishing a non-arbitrary value for this parameter.

- **Character Quantum:** The character quantum defines the cost of inputting and recognizing a letter. In the current prototype, this quantum is oversimplistic in that it is arbitrarily set to be an order of magnitude larger than the epsilon quantum, and also because it is uniform across letters, which does not seem to be psychologically plausible.

- **Decay:** Both psychological and biological evidence suggests the importance of decay. From a linguistic viewpoint, it is an essential facet of the convergence problem (Graesser and Clark, 1985). Experiments suggest leaving it at a rate of zero until all other parameters have been 'tweaked' to the user's satisfaction. This will make for traces whose sequences of events are a lot simpler to follow. Once an acceptable interpretation is generated, the user may want to introduce decay in order to find out which conceptual links are more tenuous, that is, which links will be removed from the interpretation because of decay. Only truly exponential decay is psychologically plausible but experiments demonstrate that, though 'slow' decay is basically insignificant, 'fast' decay has dramatic repercussions on the final interpretation. Indeed, 'fast' decay seems to implicitly define a time-span upper bound for all memory processes in that clusters become unreachable far more quickly, thus significantly constraining the interpretative process.

- **Working Memory Capacity:** This capacity limit defines the number of 'instantaneously' reachable clusters (section 3.2). A minimum value of 1 is required in order to handle retrievals correctly. Though implausible, a high limit (*e.g.,* more than 50 percent of STM) will not greatly affect understanding because, in the current prototype, access of clusters in STM is only one order of magnitude slower than access in Working Memory. Further investigation of the role of Working Memory, but also consideration of the 'size' of clusters is required before suggesting a plausible value for this parameter.

- **Short-Term Memory Capacity:** Because in the current prototype access to LTM is considerably 'slower' than access to STM, the capacity limit of STM seems to define, much like decay, an upper bound on all memory processes inasmuch as once in LTM, clusters become typically unreachable. Experiments suggest that a limit of less than 25 clusters does not allow for the interpretation of a short 'simple' sentence. Conversely, a psychologically implausible, extremely large STM (*e.g.,* a few hundred clusters) will possibly generate an 'in-depth' interpretation (*i.e.,* an interpretation in which even complex inferences have been used) but, unless qualitative data is extremely well-organized, will most likely lead to unexpected and undesirable interactions between KUs that incorrectly remain reachable. For example, an erroneous inference may be drawn between the current word and one that was encountered several sentences back. Much as with the capacity of working memory, further investigation is required in order to understand the tradeoff between solving the convergence problem and unduly limiting the interpretation (because of overly fast unreachability).

During experiments with the current prototype, several configurations of parameters were systematically tried with the input texts. All assumed, for simplicity, an insignificant rate of decay and an STM of 50 clusters. Separate experiments were conducted to assess the repercussions of variations of decay rate and of STM capacity on understanding. These repercussions have been briefly discussed above. Let me now summarize my observations regarding the 'standard' configurations of parameters I used with the examples:

- **Reader 1** has an 'average' KB (*i.e.*, a KB that does not includes KUs for some of the complex inferences required in some of the examples) and is given 'average' time (in terms of time-spans of races). This configuration generated the interpretation against which those resulting from more extreme configurations of parameters were compared.

- **Reader 2** has the same KB as reader 1 but extremely short races. This reader generated interpretations that lacked all of the more 'time-consuming' conceptual relationships such as causal inferences. In essence, this reader could barely do more than the simplest parsing.

- **Reader 3** has the same KB as reader 1 and long races. This reader generated interpretations that were similar to those of reader 1, mostly because there were no complex inferences in the KB to be detected, regardless of the time available. Furthermore, the longer races led to the discovery of some undesirable interactions between KUs that do not occur with reader 1.

- **Reader 4** has a 'poor' KB (*i.e.*, a KB with only the simplest parsing rules). Regardless of the values of the memory parameters. the interpretations generated by this reader were extremely superficial and incomplete with respect to those of reader 1.

- **Reader 5** has a 'rich' KB (*i.e.*, one that includes complex inferences) and is given 'average' time. Because of lack of time, the 'richness' of the KB is not systematically exploited and interpretations are typically similar or marginally more 'complete' than those produced by reader 1.

- **Reader 6** has a rich KB and extremely short races. This reader is identical to reader 2.

- **Reader 7** has the same KB as reader 5 and long races. This reader generates the most 'complete' interpretations of all the readers, but also required the most 'rule tweaking' because of unwanted interactions. Also recall that arbitrarily long races violate the initial assumption that comprehension is a real-time process.

A more extensive discussion of the results lies beyond the scope of this work because it involves complex psychological phenomena such as the use of causal inferences (van der Meer, 1987) and loss of surface information (Gernsbacher, 1985). Moreover, there are still too many unanswered questions to suggest what 'good' values for the different parameters of the model may be. But experiments with the current prototype have confirmed the importance of quantitative time with respect to a reader-based approach to understanding.

Part II

A Model of Reader-Based Comprehension

Let us briefly review part 1. In the framework of the conduit metaphor that underlies existing approaches to text comprehension, it is postulated that a text has a single determinate meaning, which is obtained by a competent reader, that is, one who possesses the correct set of *a priori* codes (or rules) of interpretation. I have argued that this strategy is normative and self-validating. Instead, I have proposed a reader-based approach to linguistic comprehension, one where 'meaning' is constituted by the interaction between text and reader. This reader is never seen as an idealized (*i.e.*, 'competent') entity, but rather as an *individual* with all the idiosyncrasies implied by that term. Choosing reader-based hermeneutics represents a significant shift in concern from existing approaches to natural language processing. In particular, the quest for the 'correct' set of rules of interpretation must be abandoned: there is no correct, or optimal, or near-optimal 'understanding algorithm'. Following this approach, I have developed, in chapter 3, a model of memory that is designed independently of semantic considerations, although the data it manipulates is retrieved from the idiosyncratic 'knowledge' base assembled and *exclusively controlled* by a specific user of **IDIoT**, the proposed comprehension tool.

In chapters 4 to 9, I demonstrate the use of the proposed model of memory and data specification tool for reader-based text comprehension. It must be emphasized that the user of **IDIoT** is the sole judge of the adequacy of the information placed in memory. Therefore, my goal is not to establish *what* 'knowledge' (*i.e.*, qualitative data) must be defined, but rather to consider the tasks generally taken to underlie text interpretation and to explain:

- How the typical rules assumed for these tasks *could* be captured with **IDIoT**. In other words, all examples of this part of the thesis are strictly illustrative and no claim is made with respect to some arbitrary 'correctness' criterion. Also, all examples only focus on the aspects relevant to the problem at hand.

- How the time-constrained quantitative processes assumed for memory correlate with the *modus operandi* of these tasks.

In other words, throughout part 2, I want to demonstrate that **IDIoT** can be used to specify a conceptual analyzer and, also, that quantitative time plays an omnipresent role in comprehension. The focus is not on the rules themselves, but rather on *how* they can be specified in **IDIoT** and *how* they interact with time-constrained memory in order to construct an interpretation of the input text.

Finally, since I postulate that the perception of subject matter cannot be grounded solely in the text, but also involves the *horizon of the reader*, I address, at the end of chapter 9, some of the idiosyncratic facets of the act of reading that I take to significantly affect comprehension.

Chapter 4

Three Models of Comprehension

Existing research in text linguistics was reviewed in subsection 2.2.2. I want to now focus more extensively on three archetypical models, from which I will establish the list of problems to be addressed in the rest of the thesis. I emphasize that these linguistic considerations were deliberately ignored when designing the quantitative processes of the time-constrained memory of **IDIoT**.

4.1 Dyer's Thematic Abstraction Units

Dyer's (1983) BORIS system uses a demon-based control structure to match many different classes of knowledge structures in a text, corresponding to different types of inferences. Each input must search memory in order to explain itself, rather than wait for some top-down process to interpret it. BORIS first builds a conceptual dependency (CD) representation (*ibid.*, pp.379–382) of the current proposition in the working memory by accessing the lexicon. The selected lexical entries or the current configuration of the CD representations may trigger the recognition of a higher level of representation for the current input. For example, the word 'restaurant' will trigger the 'restaurant' scenario, the 'restaurant' MOP (memory organization packet), and a service triangle knowledge structure. In total, BORIS has seventeen knowledge structures and twenty-eight legal interactions between them. Each link between two knowledge structures corresponds to a group of demon processes: whenever a knowledge structure is recognized (*i.e.*, matched), demons are spawned for each of its links. If a demon fires (*i.e.*, has its preconditions satisfied), then the old representation is reinterpreted in terms of the new one. This strategy allows BORIS to infer multiple connections between concepts in the story. In other words, there are no principles (such as Wilensky's (1983b) 'story understanding principles'[1], see section 2.2.2) that constrain the inferencing process. However, Dyer's set of heterogeneous knowledge structures and associated links have been criticized as being *ad hoc*, incomplete, and incorrect (Norvig, 1987, p.38).

For Dyer, 'understanding' is organized around situations in which 'failures' occur due to error(s) in planning. Those situations are represented in thematic abstraction units (TAUs), each of which represents a possible failure and its outcome. TAUs organize cross-contextual episodes that involve similar failures in planning: provided the correct TAU is selected, BORIS is 'reminded' of old expectations from previous similar narratives. More precisely, a TAU consists of the plan used, its intended effect, the reasons for failure, and knowledge of how to avoid or recover from this type of failure. Goal and expectation failures and planning choices may trigger the matching of TAUs. To decide whether or not a planning failure has occurred, Dyer proposes eleven planning metrics (*e.g.*, RISK, VULNERABILITY, LEGITIMACY, etc.). When a character in a story selects a plan, each of these metrics is checked against an *a priori* norm, and the resulting analysis determines the presence or absence of a planning error. For example, if John owes money and can either take out a loan or play Russian roulette to pay it back, a planning error will be detected, under the RISK metric, if he elects to go for the second alternative. The determination of these metrics is context-sensitive in that it highly relies on how much of the story has been processed up to that point in time. This approach is taken to be more powerful than strict metaplanning (see Wilensky, 1983b): the inherent trade-offs are defined within the narrative rather than as general principles; a bad solution can sometimes be the only alternative.

'Stories' are assumed to deal with characters and their reactions to events. Dyer claims that affective reactions of the characters reveal the underlying goal situations at an abstract level. Therefore, since BORIS must track these goal situations, it must somehow represent 'emotions'. For this task, Dyer suggests an AFFECT structure (1983, chapter 4), which allows the (positive or negative) emotional affective state of a character to be represented. More precisely, as with the detection of a planning error, *a priori* rules define how an

[1] Namely, the principles of coherence, concretion, least commitment, exhaustion, parsimony, and poignancy. Norvig (1987, p.40) remarks that "the difficulty with applying these principles to any particular text is that they contradict each other, and it is never clear how to resolve the contradictions."

affective state can be computed by comparing *metrics* to *norms*: complex psychological phenomena, and their bodily facets, are reduced to a simple *a priori* computation by *ad hoc* metrics.

4.2 Norvig's Unified Theory of Inference for Text Understanding

Peter Norvig (1987, 1989) rejects models, such as Dyer's, that concentrate on particular types of knowledge structures (such as TAUs) and construct an algorithm fitted to process those particular types of structures. Instead, Norvig advocates a unified approach to inference, one, similar to mine, in which the complexity is shifted from the algorithm to the knowledge base. This approach is formalized in a program named FAUSTUS, which is capable of drawing the *proper* inferences, that is, the ones intended by the author of the text, and avoiding improper ones. A proper inference is taken to be an assertion that is implicit, plausible , relevant, and 'easy' (*i.e.*, obtained without conscious effort). It is also stated that a 'suitable' KB is a prerequisite to making proper inferences. Consequently, a distinction is claimed between proper and idiosyncratic inferences. Such a dichotomy is quite irreconcilable with reader-based hermeneutics, since the notions of plausibility, relevance, and ease are relative to the reader, as Norvig admits himself (*ibid.*, p.3).

When FAUSTUS is given a text, input sentences are first converted to representations by a conceptual analyzer or by hand. The understanding component, FAUSTUS, takes in representations and immediately stores them in a story memory. In addition, it makes inferences, based on what is known about the story so far, as well as what is in the general knowledge base. More precisely, FAUSTUS's algorithm (*ibid.*, pp.8–9) consists of the following steps:

- **Step 0**: Construct a knowledge base.
- **Step 1**: Construct a semantic representation of the next piece of the text.
- **Step 2**: Pass markers (see subsection 2.2.2) from each concept in the semantic representation of the input text to adjacent nodes, following along the links in the semantic network.
- **Step 3**: Suggest inferences on the basis of marker collisions. For each collision, look

at the sequence of links along which markers were passed. Each link has a primitive link type associated with it, and the list of primitive link types determines the shape of the marker path that led to the collision. If the total path shape matches one of the pre-specified shapes (see below), then an inference is suggested by placing the path in the agenda.

- **Step 4:** Evaluate the potential inferences that are on the agenda. The result is either to make the suggested inference, to reject it, or to defer the decision by keeping the suggestion on the agenda. If there is explicit contradictory evidence, an inference can be rejected immediately. If several potential inferences are competing with one another, as when there are several possible referents for a pronoun, then the decision is deferred if none of them is more plausible than the others. If there is no reason to reject or defer, then the suggested inference is accepted, and new concepts are added to the model of the text.
- **Step 5:** Repeat steps 1 to 4 for each piece of the text.
- **Step 6:** At the end of the text there may be some suggested inferences remaining on the agenda. Evaluate them to see, as in step 4, if they lead to any more inferences.

Unlike the Conceptual Dependency primitives that underlie Dyer's Thematic Abstraction Units, KODIAK (Wilensky, 1986), the representation language used by Norvig, is both language- and application-independent and has no semantic primitives *per se*. Representations in the KODIAK language are composed of instances of three types of primitive objects and eight primitive associations between objects. The primitive object types are (*ibid.*, p.59):

- **Absolutes** – concepts, *e.g.*, person, action, purple, government.
- **Relations** – relations between concepts, *e.g.*, actor-of-action.
- **Aspectuals** – formal parameters for the relations, *e.g.*, actor.

The primitive link types are:

- **Dominate** – a concept is a subclass of another class.
- **Instance** – a concept is an instance of some class.
- **View** – a concept can be seen as another class.
- **Constrain** – fillers of an aspectual must be of some class.
- **Argument** – associates aspectuals with a relation.

- **Fill** – an aspectual refers to some absolute.
- **Equate** – two concepts are co-referential.
- **Differ** – two concepts are not co-referential.

From these eight types of links, five different path shapes and six inference classes are established (*ibid.*, p.101). The path shapes are regular expressions (in the sense used in formal language and automata theory) that each specify a valid sequence of links. An inference class consists of a pair of path shapes that must be matched to the two halves of the total path of a marker collision.

Throughout his dissertation, Norvig insists on the fact (*ibid.*, p.7) that:

> Declarative knowledge, when organized properly, can be used in several ways, while procedural knowledge by definition can only be used one way.... Because the knowledge base and the possible inferences are in a declarative form, it is relatively easy to combine them, to consider several inferences at the same time (as when two or more possible inferences each suggest a referent for the same pronoun). If the knowledge needed to make inferences were represented procedurally, it would be more difficult to inspect, compare, and merge inferences together. If the procedures were going to have any interaction, they would have to be written as co-routines, and would have to know some of the details of other procedures. This is often confusing and difficult, and would probably require the knowledge base modeler to modify existing inference rules to interact with new rules as they are added.... The complexity has not disappeared; it has just moved from the algorithm to the knowledge base.

Reader-based hermeneutics leads to an extreme version of this strategy: in **IDIoT** *all* qualitative data required for an interpretation is specified by the user.

Another important point made by Norvig (*ibid.*, p.77) is that views, that is, the links that support the idea of viewing one concept as another[2], need to be represented explicitly, rather than have the modeler rely on some general processing mechanism that would derive such analogical or figurative interpretations. However, such general mechanisms for mapping one concept into another have been proposed by several researchers in philosophy, psychology, and artificial intelligence (see Mac Cormac, 1985). Much like the debate between Small's (1980, 1983) word expert parsing and general parsing mechanisms, this is a 'religious' debate (Hirst, 1987, section 4.2.4), which ultimately depends on one's belief in the existence of general rules.

Views may be chained. Consider, for example, the sentence (Norvig, 1987, p.76):

[2] Norvig acknowledges the fact that several AI systems provide support for such an idea and claims that "KODIAK is the first representational language that [he knows] of to have [as] a design goal the explicit representation of views that prescribe how certain concepts can be interpreted as others."

Example 4.2.1 *The Kremlin took offense at Reagan's latest remarks.*

The word 'Kremlin' should be explicitly mapped to the concept 'soviet-government' using the view link 'place-for-organization', and 'soviet-government' to 'soviet-leaders' by applying the 'organization-for-people' view, rather than having a single general rule that would create this conceptual connection.

A major drawback of FAUSTUS is that it fails for *ambiguous* input, though it may succeed for *vague* sentences. Norvig explains this distinction by considering the sentence:

Example 4.2.2 *They saw her duck.*

He suggests (*ibid.*, p.79) that the sentence is ambiguous inasmuch as there are two possible interpretations for it:

- They saw a water fowl belonging to some female.
- They saw a female quickly bow.

And, even if we make the assumption that "*duck* refers to a water fowl[,] the sentence is still vague, in that it does not specify if the duck is male or female, large or small, alive or cooked." (*ibid.*). In other words, from my understanding, a concept is vague if some of its attributes are left unspecified. Norvig blames FAUSTUS's strict pipeline process of syntactic analysis first, inference second, for this limitation, and acknowledges that the representation of an ambiguous input was bypassed by typically hand-coding his "best guess as to what the semantic analysis would have been, had [the parser] been able to make the correct choice" (*ibid.*, p.80).

Another important limitation of FAUSTUS, and of spreading activation models in general, is that the number of possible inferences dramatically increases with the complexity of the knowledge base. Charniak (1986b) suggests not passing markers to nodes that exceed a maximal number of links. Instead, Norvig adopts the *dynamic anti-promiscuity solution* (*ibid.*, p.98):

> First run the algorithm on a representative sample of texts. Then count the markers that accumulate at each concept, and declare the *m* concepts with the most markers as promiscuous concepts.... Both solutions have an element of arbitrariness.

Also, FAUSTUS is based on the assumption that the input will be coherent. The program does not generate expectations, has no forward or top-down inferencing, and does not tackle the problem of global coherence. The important conclusion of Norvig's research is that "both script- and goal-based processing can be reproduced by a system that has no explicit processing mechanism aimed at one type of story or another, but just looks for connections in the input as they relate to what is known in memory" (*ibid.*, p.139).

4.3 Graesser and Clark's Model of Comprehension

I previously remarked that, typically, the few schema-based models that address the problem of global coherence specify a small set of *a priori* macrostructures. In contrast, Graesser and Clark (hereafter, G&C) (1985) have recently presented an extensive model of comprehension that focuses mainly on the generation and management of *bridging inferences*. Though they do not directly tackle the problem of the perception of subject matter, they suggest, much like Kintsch and van Dijk (1978, 1983), that coherence relies on, among other things, a schema for the narrative 'genre'. The existence of such a macrostructure is briefly justified (G&C, 1985, p.249):

> Adults have knowledge about the general composition of narrative passages. A typical story starts out with a description of the setting, including information about characters, the time frame, the spatial scenario, and some background episodes. Then the story proceeds with the plot. The order of episodes usually follows the chronological order of the episodes that the characters enact. Stories usually have a point which may be summarized in the form of an adage.

This obvious dependence on an 'omniscient' (with respect to the set of possible structures of a text) reader (which explains the necessity of the word 'adult') directly proceeds from earlier schema-based approaches to subject matter such as story grammars and thematic abstraction units. Indeed, G&C claim that their model includes most of the previous research done on the comprehension of *simple narratives*. It is not my intent to discuss at length their (strictly cognitive) model of understanding. However, since their model is very general (mostly, as with Kintsch and van Dijk's work (1978, 1983), because it is not implemented computationally), in this section I focus on some of its fundamental characteristics.

G&C propose procedures to model comprehension, recall, summarization, and question answering. These procedures work on *generic knowledge structures* (GKSs) represented by *conceptual graphs* (see Sowa, 1984) that they traverse and *match* in order to generate the bridging inferences that make a text locally coherent, as well as other inferences (called *projections*) that capture the reader's *expectations*. "GKSs are structured summaries or abstractions of sets of exemplars[; each one] constitutes an atheoretical data base with general knowledge about a concept" (*ibid.*, p.34). (Due to the use of conceptual graphs, the components of a GKS are referred to as *nodes* and can be thought more or less of as propositions.) The fundamental claim made is that most inferences produced during comprehension match information contained in those GKSs. Finally, in order to minimize the role of the components left out (*e.g.*, reader's horizon, language, global coherence), G&C must resort to passages that are short (less than 15 lines and 10 propositions) and 'simple' (read 'artificial'). Thus, like other information-processing approaches, non-artificial text and its possible differences in interpretation are banished in order to discover processing regularities.

The foremost characteristic of the model is the claim that (*ibid.*, p.41):

> Narrative comprehension is best viewed as a mechanism which dynamically composes a knowledge structure from nodes and structured chains that already exist in the associated GKSs. Existing GKSs "rub up against each other" in systematic ways and eventually converge on a reduced set of nodes which end up being the passage inferences.

In other words, comprehension is a *dynamic* process that *constructs a representation* of what is understood. This construction process does not reduce (at least in theory) to the *exact* matching of *a* predefined schema, but instead results from the *interactions* between several GKSs. From this viewpoint, G&C's proposal is mostly a schema-matching model with some limited inference-chaining abilities. What sets their work apart is the fact that they address the *convergence problem*, that is, they provide a description of the mechanisms they assume in order to explain how a small set of adequate bridging inferences is eventually generated.[3] Most importantly, their solution is plausible from a psychological point of view, for it is primarily rooted in the dichotomy between a *working memory* and a less accessible 'secondary memory' (*ibid.*):

> Our model of comprehension adopts the distinction between working memory and secondary memory. Secondary memory is a vast storehouse of specific knowledge structures, generic knowledge structures, and active symbolic procedures. Working memory is a limited capacity workspace where procedures and processes are executed when a person interacts with the world. For example, the process of constructing a passage

[3] As with the vast majority of information-processing models, the issue of the algorithmic complexity of the proposed model is not discussed.

> structure during comprehension takes place in working memory; the new passage structure is eventually passed to secondary memory.... Given that working memory is limited in capacity, there are limits to the number of GKSs that can occupy working memory at any point in time.

Specifically, G&C assume that at a given point in time, the reader processes one clause (in the intuitive linguistic sense of the word); all the GKSs associated with that clause are said to be *active* in the working memory. To justify the activation (and the transfer to the working memory) of *all* these GKSs, G&C observe that "a GKS is usually an automatized, overlearned conceptualization, at least those GKSs that are relevant to simple narrative passages" (*ibid.*). In other words, familiarity has bound the GKSs associated with a concept together, and thus they are accessed as a unit. G&C also assume that both the GKSs of the previous clause and a 'topic' list (which contains automatically 'recycled' GKSs such as those concerning spatio-temporal or 'genre' information (*ibid.*, p.198)) are kept in the *foreground* of the working memory. Finally, the structures obtained for the other clauses and 'episodes' processed so far are stored in the *background*, which is separate from the working memory.

Comprehension consists in constructing both bridging inferences and 'some' projection inferences between the active GKSs and the ones in the foreground or, on very few occasions, the ones in the background. Convergence relies mostly on the content and capacity constraints of the working memory to limit the generation of inferences: "comprehension suffers when the comprehender is expected to reinstate explicit propositions and inferences from several clauses earlier" (*ibid.*, p.42). In other words, the strictest linear coherence between clauses is required.

A second source of convergence comes from the fact that the GKSs have intersecting nodes: G&C claim that 90 percent of the bridging inferences involve nodes that intersect or are close to an intersection between one active GKS and another GKS. In other words, convergence is possible because the reader possesses all the GKSs necessary to match or chain the current clause to some existing knowledge.

The third and final source of convergence involves text coherence and bridging inferences. It is assumed that bridging inferences are generated during comprehension and that the projection inferences resulting from the reader's expectations play only a minor role at reading time.

Now that we have enumerated the principal assumptions of the model, let me quote at length G&C's description of the comprehension process (*ibid.*, p.43):

> When an incoming clause N is interpreted, the old passage structure (created from clauses 1 to $N - 1$) is modified and a new structure is constructed. According to our model, four procedures accomplish this modification. These four procedures may be accomplished in the order that we list and describe them. During procedure 1, a check is made for a direct match between clause N and an existing node in the old structure. For example, a match occurs when a prior expectation is confirmed by the incoming clause N. During procedure 2, a set of bridges are formed between clause N and the old passage structure in working memory. When a bridge cannot be formed, the comprehender attempts to reinstate old passage nodes in working memory and then construct a set of bridges. Alternatively, the comprehender concludes that a new topic or episode is introduced. After the bridging procedure is executed, the pruning procedure is executed (procedure 3). The erroneous nodes in the old structure are deleted in light of new information and constraints posted in working memory. During procedure 4, projection nodes are constructed. The projection nodes include expectations and elaborations. The construction of projection nodes is particularly sensitive to the comprehender's goals and many of these nodes may not be constructed during comprehension. When the four procedures are considered as a whole, we refer to the construction mechanism as the *matching-bridging-pruning-projection* mechanism (abbreviated MBPP).

The final structure produced by the MBPP algorithm consists of a set of separate and distinct 'episodes'; the scope of an episode entirely depends on the linear coherence assumed between the clauses of a text. Since the model is not implemented, it is difficult to grasp the exact nature and scope of an episode. It appears that the final representation of each episode consists of *all* the GKSs and the bridging inferences that were still active at the end of the processing of that episode. This aspect of the model is not explicitly dealt with since, from my standpoint, the four simplistic passages used as examples each consist of a single episode. Thus, I repeat, G&C's model can only account for the strictest linear form of clausal coherence.

It is clear that the MBPP algorithm does not deal with the perception of subject matter. Indeed, this problem is postponed to retrieval time: the reader is assumed to *reconstruct* subject matter only when asked to recall, summarize, or answer questions about the narrative. This reconstruction process assumes an *a priori* macrostructure for the text and a *central content selector* that traverses the (extremely rich) passage structure obtained (for each episode) at comprehension time and selects nodes and bridging inferences with respect to their content. Although this selection is content-dependent, G&C assume the existence of three general principles that underlie this process:

- Principle of inferability: typically, information that can be inferred or reconstructed is not selected (Kintsch and van Dijk, 1978, 1983).

- Principle of structural centrality: *pivotal* information (Lehnert, 1983) is selected.

- Principle of pragmatic communication: information that conveys a major message or point (Wilensky, 1982, 1983a) is selected.

In summary, G&C's model of comprehension consists in the generation and pruning of a multitude of bridging inferences in working memory.

4.4 The Problems of Text Interpretation

The models of understanding summarized in the previous sections introduce most of the typical issues that must be addressed by a theory of linguistic comprehension. In this section, I wish to establish a list of some of these issues.

First, I must remark that none of the reviewed models draw on work in formal semantics, a field that currently occupies a large portion of the research spectrum in philosophy and artificial intelligence. Indeed, the relevance of formal semantics to text understanding is generally rejected (*e.g.,* Norvig, 1987, p.20; Graesser and Clark, 1985, p.23). In fact, several researchers (*e.g.,* Minksy, 1975; Braine, 1978; Johnson-Laird, 1982) emphasize the inadequateness of logical formalization with respect to the fundamental goal of cognitive science, that is, the understanding (by modelling) of cognition: formal reasoning is taken to account only for a small portion of adult thinking. From my standpoint, 'human reasoning' seems to consist in the ability to apply an inferential schema that specifies a rule defined over abstract symbols. The immediacy of the application of such a rule depends on:

- The individual's ability to abstract.

- The intellectual resources that may be brought into play by an individual in order to apply the rule. Intuitively, it appears that the more abstract (or complex) a mental operation is, the more memory resources it requires (up to the point where pencil and paper become necessary), and the more time it takes to execute (possibly due to the hypothesis that the manipulation of abstract symbols is inherently sequential).

- The retrievability of the rule itself.

In other words, I suggest that the ease and frequency of use of reasoning techniques can be somewhat idiosyncratic. Since reading does not allow, in general, sufficient time to make such complex inferences (see Garrod, 1985), I will not investigate this topic further.

Second, certain hypotheses of the reviewed models are not relevant to hermeneutic considerations, but rather to the underlying representational scheme:

- Though FAUSTUS (Norvig, 1987) does not handle expectations, they are typically assumed to play a role in comprehension. For Graesser and Clark (1985), projections, which result from the reader's expectations, are extremely idiosyncratic with respect to the reader and thus constitute only a minor facet of their MBPP algorithm. Conversely, expectations are omnipresent in BORIS (Dyer, 1983). They are crucial, for example, for the detection of a planning error or the determination of an affective status. In **IDIoT**, expectations are taken to constitute a quantitative phenomenon that is directly modelled in the proposed time-constrained memory, rather than tackled at the qualitative level.

- The problem of convergence, which is typically ignored by schema-matching approaches to text understanding but central to Graesser and Clark's discourse, does not pertain to text interpretation so much as to a model of memory. Indeed, both the working memory constraints and the dichotomy between a foreground and a background in this working memory that Graesser and Clark postulate correlate to a discussion of the organization and management of memory. Recall that, in **IDIoT**, the partitioning of memory follows a temporal distinction rooted in the retrievability of the memory elements.

- The complex markers and very sequential evaluation step hypothesized by inference-chaining models such as FAUSTUS implement another form of a convergence strategy, one that implies the transmission of complex data structures across a network of memory elements, as well as a procedural component that manipulates these structures and seems to operate *in vacuo*, that is, completely outside of the assumed spreading activation memory architecture. Such an approach, which violates the biological constraint (see subsection 2.4.2), is rooted in a particular model of memory that ought to be, from a reader-based viewpoint, designed independently of any hermeneutic consideration, as in chapter 3. Recall that, according to the proposed time-constrained memory, there is no need for complex markers or for the *ad hoc* anti-promiscuity rules of inference-chaining models (*e.g.*, Charniak, 1986b; Hendler, 1987, 1989; Norvig, 1987, p.97).

- FAUSTUS and BORIS are both limited to bottom-up processing: an inference can only be made if all its premises are present. In **IDIoT**, however, a mechanism has been introduced to allow for both forward and backward chaining of inferences (see section 3.4.1). The point is that the *modus operandi* of the inference engine should be discussed at a strictly quantitative level.

IDIoT proceeds from the 'trivial algorithm' design philosophy (see chapters 1 and 2) that stems from a reader-based approach to text interpretation. Such an approach can be viewed as Norvig's design philosophy carried to its extreme: no semantics are to be statically encoded in procedures, all the qualitative complexity of comprehension is shifted to the knowledge base. My goal in the rest of this dissertation is to illustrate how some of the typical rules and knowledge structures assumed in the reviewed models *could* be coded using the suggested representational scheme.

An investigation of text comprehension cannot, however, simply consist in explaining how existing models can be almost entirely 'de-proceduralized'. There are several specific qualitative problems that must be addressed:

- The role of grammar, which is significantly downplayed, if not entirely bypassed, in the surveyed models[4], is too central to modern linguistics to be ignored.

- Similarly the general problem of disambiguation (Hirst, 1987), which involves, at least, the issues of lexical and structural disambiguation, must be considered at length, for it constitutes an essential part of comprehension. Indeed, from my standpoint, it appears disambiguation is the most pervasive task of linguistic comprehension at the clausal and sentential levels.

- It is a common view that inferencing is central to linguistic comprehension (Kass, 1986). Although I abandon Norvig's notion of a 'proper' inference (as opposed to an idiosyncratic inference), I must explain how bridging inferences are drawn during

[4] Like most 'semantic' parsers, Dyer's McDYPAR parser (1983, Appendix III) has some serious limitations (Marcus, 1984), and parsing is not addressed at all by Norvig or by Graesser and Clark.

reading. A fundamental characteristic of **IDIoT** is that the quantitative mechanisms of the suggested time-constrained memory correlate directly to the reader's behavior in both the disambiguation and the bridging tasks.

- All of the models reviewed assumed that if contradictory evidence exists for a potential inference, then the inference should be rejected. Human reading is not that simple. For example, Granger and Holbrook (1983) investigate the notion of an 'inferencing strategy' to manage the semantic conflicts triggered by contradictory evidence. I shall suggest how such a strategy may be specified in **IDIoT**.
- As mentioned in chapter 2, some approaches to text interpretation often rely on the intuitive notions of 'fact', 'event', and 'episode'. For completeness, the perception of such categories must be discussed.
- Within the framework of reader-based comprehension, I shall also focus on other issues that I assume to account, in part, for the idiosyncratic nature of comprehension.

A recurring theme of this part of the thesis is that the *modus operandi* of time-constrained memory underlies almost all facets of the proposed model of text interpretation, and ultimately, the perception of subject matter. In particular, the suggested model of memory offers a solution to the convergence problem: through the management of STM and of semantic conflicts, and, most importantly, because of the role of time as a stopping criterion for inferencing, only a limited number of clusters are constructed during reading. The set of clusters constructed during the reading of the text constitutes the output of **IDIoT**, the interpretation that the user can examine. Thus, I emphasize, the perception of subject matter is *not* postponed to (post-reading) retrieval time, as assumed by several researchers, but instead directly results from this convergence on a limited set of inferences. Any reconstruction that would result from *a posteriori* tasks such as recall, summarization, or question answering, is *not* considered in this research.

4.5 Preliminaries for Clausal Interpretation

A text consists of a sequence of sentences. In turn, each sentence is composed of one or more *clauses*. Recall that most research in NLP has focused on the artificial understanding of a single sentence *in vacuo*, that is, as a linguistic object deprived of any context. The processing of larger linguistic units has generally remained a stumbling block (Habel, 1983). Within the sentential framework, it is typically assumed that a sentence is first analyzed syntactically, then the process of *semantic interpretation* maps the resulting parse tree to a representation of the sentence's 'meaning'. Some researchers (*e.g.*, Schank and Birnbaum, 1984; Birnbaum, 1989) have argued against such undue emphasis on syntax in isolation and instead advocated a more semantic and integrated set of rules of interpretation. Also, recall that the existence of rules altogether has been attacked by connectionist researchers (*e.g.*, Rumelhart, 1984). Finally, I remark that existing computational models of NLP can account only with difficulty for basic psychological phenomena such as the loss of surface information (see Gernsbacher, 1985).

In the following chapters, I will discuss the linguistic problems generally considered when addressing the comprehension of a few clauses and sentences. First, in chapter 5, the problem of syntax is considered in order to present **IDIoT**'s unified approach to clausal interpretation. Then, the general problems of referential resolution, lexical and structural disambiguation are studied at length respectively in chapters 6, 7, and 8. The problems of idioms and 'non-literal' interpretation are very briefly investigated at the end of chapter 7. The issue of figurative language requires an extensive discussion that lies beyond the scope of this dissertation. In chapter 9, I discuss bridging inferences in **IDIoT**, and conclude with a recapitulation of the linguistic issues addressed in part 2 of the thesis.

Finally, I remark that in the current prototype for **IDIoT**, the words of the input text are treated as innate features, that is, features that are automatically detected from the presence of a string of characters in the input (see chapter 3). The task of word recognition, which is typically assumed to involve the issues of phonology and morphology, is therefore bypassed. However, it appears that the proposed model of memory is especially well-suited for Norris's (1986) 'checking model', which is based on the notion of multiple candidacies over a short interval of time.

Chapter 5

On Grammar and Syntax

5.1 Syntax with IDIoT

The psychological reality of a linguistic grammar and of a parsing mechanism is fairly well established. Regardless, several researchers have criticized the relevance for linguistic comprehension of the conceptualization of a grammar as a set of consistent rules defining the well-formedness of a parse tree: as explained when story grammars were discussed (subsection 2.2.1), text interpretation is *not* an issue of well-formedness. Moreover, existing theories of grammar are typically grounded in the conduit metaphor and its implied inaccessible (and self-validating) notions of competence and language, which have been abandoned in favor of reader-based comprehension. For example, Lindsay and Manaster-Ramer (1987) write:

> Almost all computational work on natural languages, in and out of AI, has adopted the conception of language, derived from traditional grammar and structural linguistics, according to which there exists a body of knowledge which defines the primitives of which the language is composed, the principles by which these primitives may be combined, and the meanings associated with each primitive and with each principle of composition. This conceptual homogeneity has been obscured by the controversies surrounding virtually every other question in NL processing, such as the debate over models which postulate syntactic and/or logical levels of analysis in addition to the meaning (conceptual) representation,... as opposed to models which relate utterances to meanings in an integrated fashion,... which has sometimes been viewed as involving a contrast between grammar-based and knowledge-based methodologies.... [A]lmost all models purport to simulate the idealized user of a given language, conceived as an expert on his native language,... and consequently assume that there is such a thing as a definite knowledge of *what* a native speaker does in the way of assigning some structural representations.

It should be clear from the discussion of chapter 2, that I discard the notions of linguistic competence of an idealized user, and of a correct set of rules of understanding, and thus, of a grammar as a set of consistent rules of composition. In this thesis, I focus on the idiosyncratic performance of a reader. That, in essence, I favor user-specified 'word experts' (see chapter 2 and Small, 1980, 1983) for clausal interpretation, as shall become evident in the following chapters, does not modify this position: my technical standpoints regarding the form and organization of qualitative data do not alter the fact that I abandon the notion of linguistic competence, that is, the existence of a correct (and definitive) corpus of knowledge for linguistic comprehension, at both the sentential and textual levels. In other words, from my viewpoint, regardless of its expression, knowledge, including grammar, is always idiosyncratic in that it is acquired, owned, and managed by a specific individual who is the sole judge of the acceptability of an interpretation. The mind's ability to often understand so-called 'ungrammatical' sentences and to misunderstand or simply not understand grammatical sentences seems to reinforce this standpoint.

Because I adopt this perspective, there's no point in reviewing existing theories of syntax (which typically ignore the arbitrarily separated 'semantics'). I do not develop a grammatical theory, and I do not consider the complexities studied by linguists. From my standpoint, grammar is not a system of consistent rules of composition, but merely an *acquired* system consisting of often-inconsistent conventions used to *constrain* the arbitrariness between signifiant and signifié that Saussure (1916) took as the fundamental principle of linguistic communication. It is essential that a majority of the users of a language share a large number of graphological, grammatical, and even lexical conventions, in order for the inherent arbitrariness of communication to be constrained, and thus for comprehension to be made possible. This sharing of conventions does not imply an innate 'competence' but merely attempts to limit the arbitrariness of reading to the 'semantic' level(s), and does not *necessarily* eliminate it at the grammatical and lexical levels. In other words, I suggest that grammar merely constitutes a conventionalized linguistic mechanism to *speed up* the recognition of certain semantic cues. Since I view comprehension as a time-constrained process for which past processing plays an important role, I feel particularly close to Lindsay and Manaster-Ramer (1987) when they explain that:

> If there is such a thing as a grammar, only parts of it will be accessed at any given point in time in real-time processing. As a result we would be able to explain how a speaker can consider a sentence ungrammatical for a time and then decide, upon further reflection (*i.e.*, when more of the grammar is considered) that it is grammatical, or vice versa.

Within the framework of **IDIoT**, any grammatical rule must be specified by means of the associations, candidates, constraints, and expansion procedures of one or more features (see chapter 3). Since knowledge units (which implement features) and their communications both consume time (to be retrieved and to transmit a signal respectively), it is possible to model rules that require a long time to retrieve and therefore, can be missed if the time-span allocated to produce an interpretation is too short, the phenomenon described by Lindsay and Manaster-Ramer. Furthermore, it is important to understand that all rules are treated equally in **IDIoT**, that is, as data for the trivial algorithm (see chapters 1 and 2). Thus, there is no need to assume separate algorithms and data structures for the processing of syntax. Indeed, the organization of short-term memory in the proposed model of time-constrained memory implements *de facto* a stack, which is often used in conventional parsers. Also, since candidacies span an interval of time after having been triggered, they can consider subsequently processed information without having to resort to counterintuitive look-ahead mechanisms. In other words, some of the characteristics hypothesized for conventional parsers can be found at the quantitative level of **IDIoT**. Most importantly, there is no need to assume that syntax is processed before semantics or *vice versa*, as the distinction between syntax, semantics, and pragmatics becomes irrelevant in the framework of reader-based comprehension, and more specifically, of a trivial algorithm. It follows that, contra Lytinen (1984), it is not even necessary to keep syntactic and semantic knowledge in separate KBs.

Let us focus on the specification of a grammatical rule using **IDIoT**. The triggering mechanism of any KU allows the user to specify the static preconditions of a rule. The possibly-ordered triggers are those features that must always be present in order for a rule to apply (*i.e.*, become detected). Through the use of the testEquivalenceOf, testDifferenceOf, testPresenceOf, and testAbsenceOf instructions, the dynamic preconditions of a rule may be checked using buildable features. For example, the candidacy of a feature, called say **forcedPP-Attach**, which attaches a locative PP (*e.g.*, 'on the beach') to a verb, may be triggered by the presence of a verb and of such a PP but may require that the verb in question does not have another locative PPattached to it at that point in time. If we assume that a locative PP is attached as a subcluster under feature **location** in the cluster **u1** associated with the verb, then **forcedPP-Attach** can verify this dynamic constraint using the instruction:

```
testAbsenceOf location in u1
```

which ensures that if feature **location** is absent from u1, then the expansion procedure of **forcedPP-Attach** proceeds; otherwise, this procedure immediately fails.

The use of weights, especially for exceptions, allows a rule to admit a certain degree of 'ungrammaticality'. For example, a constraint that attaches the subject and verb of a clause may be specified in such a way that it can become detected even if the feature **SVA** that detects a subject-verb agreement is absent (*i.e.*, has not been detected): all that is necessary is for a smaller weight (with respect to the detection threshold) to be assigned to **SVA**.

Any transformation of the current context (which consists of a set of reachable clusters) can be specified through the expansion procedure of a rule, the proposed set of instructions for such procedures allowing for all the basic data structure operations. Let me elaborate. Recall that a particular cluster can be accessed through one of the features it contains. For example, all verbs denoting an action could have feature **actionVerb** put in the cluster constructed by their expansion procedure. Then, using the instruction:

```
getCluster u1 governing actionVerb
```

a rule can access the most reachable cluster associated with an action verb, independently of the actual verb itself. Once u1 is bound to this cluster, it is possible to add features and subclusters to it, rename features, etc. Since an expansion procedure can access several clusters, it is possible for a single detected feature to arbitrarily modify all the clusters it accesses. Experimentation with the current prototype of **IDIoT** suggests that access to the relevant clusters of a rule constitutes one of the tasks that must be well thought out by the user. For example, as will be illustrated in the examples of the next section, the clusters associated with the different NPs of a sentence should contain features that precisely identify the role of each NP in the sentence: a feature that captures a rule does not want to access the direct object of the verb or the noun phrase of a PP if it is the subject NP that is relevant to the rule. Experimentation also suggests that relays (*i.e.*, KUs that simply relay signals from their suppliers to their customers; see chapter 3) are very useful to handle more 'general' rules. For example, if feature **actionVerb** is a relay with all action verbs as suppliers and all rules relevant to such verbs as customers, then each time an action verb becomes detected, all the rules that apply to action verbs in general will be notified through **actionVerb**. For example, if the word 'eat' eventually leads to the detection of **actionEat**, then all rules specific to **actionEat** will be notified because of this detection, and all rules for action verbs in general will be subsequently notified through the detection of **actionVerb** that results from the detection of **actionEat**.

It should be clear that these considerations are not limited to grammatical rules, but, in fact, hold for all rules. Thus, there is no need to postulate a separate methodology for grammar: all grammatical rules are specifiedby means of features and the parse tree, if any, is fully integrated with the clusters constructed and modified through the expansion procedure of the features which become detected during the reading. Furthermore, since rules are implemented as KUs of a time-constrained memory, parsing is necessarily a time-constrained process.

Incidentally, the importance of punctuation must be stressed with respect to such time-constrained parsing: not only does a punctuation sign require some time to process, but it may also provide a clue with regards to the end of a clause. For example, a period followed by a space and a capital can denote the end of a sentence and the start of another one. Upon detecting the end of a sentence, some of the existing candidacies could be interrupted and a particular parse adopted.

Finally, I remark that some syntactic cues such as the tense of verbs will play an important role for text interpretation (which is taken to mostly depend on the perception of change of time frame and/or location; see section 9.2) and that the phenomenon of the loss of surface information (Gernsbacher, 1985) will be very simply explained by having most syntactic rules being more retrievable, and thus more readily detectable, than complex thematic rules (again, see details in section 9.2).

5.2 Examples of Syntactic Processing with IDIoT

In **IDIoT**, several approaches to parsing are possible. For example, the use of relays and confirmation paths in the following examples could be replaced by a set of rules in which detection is possible only through the satisfaction of expectations. It is not my goal to explore the different representational strategies, but, instead, to propose an approach to some typical problems associated with syntax.

Let us start with the first steps involved in the processing of a word. For example, consider the words 'woman' and 'women', to which correspond the innate features **'woman'** and **'women'**. Let us assume that the words are associated with a single concept, namely feature **woman**. There are several different ways of specifying these features. Here are three possible sets of definitions:

Possibility 1: Using the 'empty' feature **rootWoman** to capture the morphological root shared by the two words and ignoring agreement rules:

```
innate KU 'woman':
 associations: woman
 expansion:
      getNewCluster u1
      addFeature rootWoman to u1
      addFeature singular to u1

innate KU 'women':
 associations: woman
 expansion:
      getNewCluster u1
      addFeature rootWoman to u1
      addFeature plural to u1

KU woman:
 constraint c1:
  triggers: 'woman'
  inputs:
      noun has a weight of 1
 constraint c2:
  triggers: 'women'
  inputs:
      noun has a weight of 1
 expansion:
      getCluster u1 governing rootWoman
      addFeature person to u1
      addFeature female to u1

KU rootWoman:
```

Possibility 2: Identical to possibility 1, but moving the constraints from **woman** to **rootWoman** for which the former becomes a customer port:

```
innate KU 'woman':
 associations: rootWoman
 expansion:
      getNewCluster u1
      addFeature rootWoman to u1
      addFeature singular to u1

innate KU 'women':
 associations: rootWoman
 expansion:
      getNewCluster u1
      addFeature rootWoman to u1
      addFeature plural to u1

KU woman:
 constraint c1:
  triggers: rootWoman
  inputs:
      noun has a weight of 1
 expansion:
      getCluster u1 governing rootWoman
      addFeature person to u1
      addFeature female to u1

KU rootWoman:
 constraint c1:
  triggers: 'woman'
 constraint c2:
  triggers: 'women'
 port:
      woman has a delay of 1 and is a customer
```

Possiblity 3: Using conditional instructions and eliminating **rootWoman**:

```
innate KU 'woman':
 associations: woman
 expansion:
      getNewCluster u1
      addFeature 'woman' to u1
      addFeature singular to u1

innate KU 'women':
 associations: woman
 expansion:
        getNewCluster u1
        addFeature 'women' to u1
        addFeature plural to u1

KU woman:
 constraint c1:
  triggers: 'woman'
  inputs:
        noun has a weight of 1
 constraint c2:
  triggers: 'women'
  inputs:
        noun has a weight of 1
 expansion:
        ifConstraint c1 then getCluster u1 governing 'woman'
        ifConstraint c2 then getCluster u1 governing 'women'
        addFeature person to u1
        addFeature female to u1
        addFeature noun to u1
```

Each set of possible definitions corresponds to a different level of data specificity, but also to a different sequence of exchanges of signals. In the first set, the feature **rootWoman** is detected through the expansion procedure of **'woman'** or **'women'**. More specifically, since **rootWoman** is not specified anywhere else in the definition of these innate features but in an addFeature instruction of their expansion procedure, **rootWoman** is sent a forced detection signal and immediately becomes detected upon receiving this signal. In the second set, since **rootWoman** is explicitly specified as a 'forced detection' of the innate features, it is merely sent the presence signal of the detected innate feature.[1] Upon receiving this presence signal, **rootWoman** does not immediately become detected, but instead starts its candidacy. Since the constraints of **rootWoman** have only triggers, and since it is not buildable, the triggered constraint will necessarily be immediately satisfied, thus leading to the detection of **rootWoman**. The third set of definitions illustrates the use of conditional instructions to regroup in a single expansion procedure the actions associated with the detection of a several distinct features. In this set, the clusters constructed by **'woman'** and **'women'** do not share a feature, and thus conditional accesses (*i.e.*, combinations of ifConstraint and getCluster instructions) are required in the expansion procedure of **woman** in order to access the cluster of its triggering feature.

[1] An innate feature sends a presence signal, not a forced detection signal to its associations.

In all three sets of definitions, **woman** has the limitation that it can be detected only if its syntactic category, that is, the feature **noun**, is felicitous at that point in the processing of the clause. Feature **noun** is a relay with all possible nouns as suppliers, and all general rules of valid noun usages (*e.g.*, **NP**) as customers. The latter rules capture where a noun (any noun) is felicitous in a clause: after a determiner, after an adjective, after a transitive verb, etc. Such valid sequences can be detected using ordered triggers, expectations, and, if necessary, even buildable features.

In all three sets of definitions, the problem of number agreement is ignored, number information being directly constructed by the innate features. One solution could consist in having **woman** submit **nounSingular** if triggered by **'woman'**, and **nounPlural**, if triggered by **'women'**. Such features cannot be submitted by the innate features themselves since, by definition, innate features do not have constraints. The difficulty with this approach is that the syntactic category and the number agreement of a word do not seem to have the same importance: a sentence can be intelligible even if number agreement is violated. As an alternative, **woman** could submit **noun** with a weight of 0.9, for example, and **singular** with a weight of 0.1. This approach may be more intuitive, but still has a flaw: **singular** and **noun** are treated as equals when, in fact it seems to me that the detection of **singular** should only be relevant once **noun** has been detected. In other words, number agreement first requires establishing the syntactic category of a word. In view of these observations, I suggest that agreement be handled after word recognition. Again, several strategies are possible: are agreements detected, or disagreements, or both?

A disagreement constitutes an ungrammaticality. On the one hand, some models of NLP attempt correcting these ungrammaticalities but, as remarked by Lindsay and Manaster-Ramer (1987, p.101), such corrections are hazardous if they ignore the context. On the other hand, linguists simply reject the sentence. In the case of the human mind, the reader may notice (*i.e.*, become aware of) a disagreement, possibly infer a correction or re-read the passage in question, and continue reading. From that viewpoint, a syntactic disagreement constitutes one kind of *conflict* in the interpretation. This topic will be investigated in subsection 9.4.1. The point for now is that the detection of a disagreement will lead to the detection of a conflict feature that handles such problems. Let us focus on how a disagreement may be detected. Continuing with our example, let us study number disagreement for this sentence

Example 5.2.1 *A women cries.*

I suggest the following scenario:

1. Feature **startOfSentence** is detected. This feature captures the processing of a new sentence, as explained in chapter 3.

2. The word 'A' leads to the detection of **aArticle** whose cluster contains features **singular** and **determiner**, which also become detected.

3. Feature **determiner** is a trigger of **NP**.

4. The word 'women' constructs a cluster including **plural**, and triggers **woman** which submits **noun**.

5. Feature **NP** is a customer of **noun** and thus receives a submission signal from **noun**.

6. Feature **NP** has one of its constraints consisting of the ordered triggers **determiner** and **noun** satisfied, and thus sends a confirmation signal back to **noun**, which becomes detected and sends a reinforcement signal to **NP**.

7. Upon receiving reinforcement from **noun**, **NP** becomes detected and notifies, among others for example, **numberAgreement, numberDisagreement, validNPUsage** and **invalidNPUsage**.

8. Feature **numberDisagreement** is a relay with **NP** as its supplier and **singularDisagreement** as one of its customers.

9. Feature **singularDisagreement** is a buildable feature with customer **syntacticConflict**:

```
KU singularDisagreement:
% This feature checks that a singular determiner and
% a plural noun are in the same NP.
is buildable
 associations: syntacticConflict
 constraint c1:
      triggers: NP
 expansion:
```

```
          getCluster u1 governing NP
          getCluster u2 governing determiner in NP
          testPresenceOf singular in u2
          getCluster u3 governing noun in u1
          testPresenceOf plural in u3
     ports:
          ...
```

Other disagreement features (if any) and the corresponding agreement feature(s) (if specified) are all similar to **singularDisagreement**.

10. Feature **singularDisagreement** has its expansion procedure successfully built in the current context, and thus becomes detected, leading to the perception of a conflict (*i.e.,* to the detection of **syntacticConflict**).

11. Feature **validNPUsage** is a relay with **NP** as its supplier and **validSentenceStart** as one of its customers.

12. The ordered triggers **startOfSentence, NP** satisfy a constraint of **validSentenceStart** which becomes detected.

The proposed scenario also addresses the syntactic felicity of an NP in a sentence by the features **validNPUsage** and **invalidNPUsage**. The latter could have **NPfollowingIntransitive** as a customer. In its simplest form, this feature would have a constraint with only the ordered triggers **intransitive** and **NP**. It would become detected if an intransitive verb was followed by an NP, and would lead to the perception of a syntactic conflict.

It should be noted that the handling of number agreement is designed independently of adjectives that could be inserted between the determiner and the noun. Consider, for example, the NP 'a short beautiful young women', for which I suggest the following scenario:

1. The word 'A' and its number agreement with the word 'women' are handled exactly as in the previous example.

2. The word 'short' leads to the candidacy of **short**, which submits **adjective**.

3. **adjective** is a relay with **adjectivesStart** as a customer.

4. The sequence **determiner, adjective** satisfies a constraint of **adjectivesStart**, which sends a confirmation signal to **adjective**.

5. Features **short, adjective,** and **adjectivesStart** all eventually become detected.

6. The expansion procedure of **adjective** makes the cluster associated with **short** a subcluster under feature **adjectives** in the cluster associated with the determiner.

7. The word 'beautiful' leads to the candidacy of **beautiful,** which submits **adjective.**

8. Another customer of **adjective** is **adjectivesSuite,** which has one of its constraints satisfied from the co-occurrence of **adjectiveStart** and **adjective.** Thus, **adjectivesSuite** sends a confirmation signal back to **adjective.**

9. Features **beautiful, adjective,** and **adjectivesSuite** all eventually become detected.

10. Again, the expansion procedure of **adjective** makes the cluster associated with **beautiful** another subcluster under feature **adjectives** in the cluster associated with the determiner.

11. The word 'young' is processed like the previous adjectives, with the only difference being that **adjectivesSuite** would send a confirmation signal to **adjective** through the satisfaction of another of its constraints with only the ordered triggers **adjectivesSuite, adjective.** In other words, the sequencing of adjectives is handled by a recursive constraint.

12. Upon the eventual detection of **NP,** the clusters associated with the determiner, the adjectives, and the noun can be reorganized into a single cluster that is easier to manipulate.

Anticipating the chapter 7 on lexical disambiguation, I suggest the following (still very incomplete) definitions[2] for the more complex case of the words 'man' and 'men' ('man' can be a noun or a verb):

```
innate KU ’man’:
% This feature handles the recognition of the word ’man’
```

[2] Possible agreement and disagreement features for verbs, with respect to person, tense, etc., are not considered.

```
 associations: nounMan, actionMan
 expansion:
      getNewCluster u1
      addFeature 'man' to u1

innate KU 'manned'
% This feature handles the recognition of the word 'manned'
% which is known to be a verb in the past form.
 associations: actionMan
 expansion:
      getNewCluster u1
      addFeature 'man' to u1
      addFeature pastForm to u1

innate KU 'mans':
% This feature handles the recognition of the word 'mans'
% which is known to be the third singular person of a verb.
 associations: actionMan
 expansion:
      getNewCluster u1
      addFeature 'man' to u1
      addFeature 3rdPersonSingularForm to u1

innate KU 'men':
% This feature handles the recognition of the word 'men'
% which is known to be a plural.
 associations: nounMan
 expansion:
      getNewCluster u1
      addFeature 'man' to u1
      addFeature plural to u1

KU nounMan:
% This feature corresponds to the concept of man as a noun.
% The different morphological forms of the concept define
% its constraints.
% If 'man' leads to the detection of nounMan,
% then actionMan can be inhibited.
  constraint 'man':
   triggers: 'man'
   inputs:
       noun has a weight of 1
   outputs:
       sends inhibitionSignal to actionMan
  constraint 'men':
   triggers: 'men'
   inputs:
       noun has a weight of 1
 expansion:
      getCluster u1 governing 'man'
      addFeature noun to u1
      addFeature person to u1
      addFeature male to u1
      ifConstraint c1 then addFeature singular to u1
 ports:
      ...

KU actionMan:
% The different morphological forms of the verb define
% its constraints. The different meanings of the word
% are the customers of this feature.
 constraint 'mans':
  triggers: 'mans'
  inputs:
      verb has a weight of 1
 constraint 'man':
  triggers: 'man'
  inputs:
      verb has a weight of 1
  outputs:
      sends inhibitionSignal to nounMan
 constraint 'manned':
  triggers: 'manned'
  inputs:
      verb has a weight of 1
      ...
 expansion:
      getCluster u1 governing 'man'
      addFeature actionMan to u1
      addFeature verb to u1
      addFeature compulsoryTransitive to u1
 ports:
      'man' has a delay of 1 and is a supplier
      'mans' has a delay of 1 and is a supplier
      'manned' has a delay of 1 and is a supplier
      manAShip has a delay of 1 and is a customer
        actionSupplyWithMen has a delay of 1 and is a customer
      ...
```

If desired, these definitions could handle more details at the expense of being more complex. For example, the expansion procedure of **nounMan** could build a more complex structure to reflect that **male** is a type of **sex**.

The problem of subject-verb agreement should probably not act as a gating factor for the detection of a subject-verb relationship, and can be treated much in the same way as number agreement: once the subject and the verb have been established, features similar to **singularDisagreement** can be used to check that the number of the subject agrees with the person of the verb. Consider the following two features for detecting subject-verb disagreements:

```
KU sva-1:
% This feature becomes detected if the subject is in the 3rd
% person and the verb is not and is not in past tense form.
is buildable
 associations: syntacticConflict
 constraint c1:
  triggers: subj-verb-rel
 expansion:
      getCluster u0 governing clause
      getCluster u1 governing subject in u0
      getCluster u2 governedBy subject in u1
      getCluster u3 governedBy NPhead in u2
      testPresenceOf 3rdPerson in u3
      testPresenceOf singular in u3
      getCluster u3 governedBy mainVerb in u1
      testAbsenceOf pastForm in u3
      testAbsenceOf 3rdPersonSingularForm in u3
 ports:
      subj-verb-rel has a delay of 1 and is a supplier

KU sva-2:
% This feature becomes detected if the verb is in 3rd person
% form, but the subject is not.
is buildable
 associations: syntacticConflict
 constraint c1:
  triggers: subj-verb-rel
 expansion:
      getCluster u0 governing clause
      getCluster u1 governing mainVerb in u0
      getCluster u2 governedBy mainVerb in u1
      testPresenceOf 3rdPersonSingForm in u2
      testPresenceOf singular in u3
      getCluster u1 governing subject in u0
      getCluster u2 governedBy subject in u1
      getCluster u3 governedBy NPhead in u2
      testAbsenceOf singular in u3
 ports:
```

```
subj-verb-rel has a delay of 1 and is a supplier
```

Syntactic disagreement (*e.g.*, subject-verb disagreement) should not prevent the establishment syntactic categories and semantic relationships; ungrammaticality should not prevent understanding. Also, the several different types of syntactic conflicts can be captured in the constraints of a few KUs, if not of a single one.

The problem of recognizing the direct object of a verb was discussed in the example of section 3.7.

For the sentence:

Example 5.2.2 *John likes.*

with no implicit direct object in context, the candidacy of **missingDirectObject** is triggered. This buildable feature, whose definition is given in section 3.4.3, will have its expansion procedure succeed and will become detected.

And for the sentence:

Example 5.2.3 *John gives the girl red roses.*

once the direct object has been detected, the sequence **actionVerb**, **directObject**, **NP** triggers **indirectObject**. Feature **indirectObject** would be a buildable feature that checks that the verb is bitransitive and already has a direct object. Upon its detection, **indirectObject** has its expansion procedure first move the existing direct object cluster under the new feature **indirectObject**, which is added to the cluster governing the verb, and then add the most reachable NP cluster (*i.e.*, 'red roses') under feature **directObject**. Here is a possible expansion procedure to achieve this:

```
getCluster u1 governing mainVerb
getCluster u2 governedBy mainVerb
testPresenceOf bitransitive in u2
testPresenceOf directObject in u1
addFeature indirectObject to u1
moveSubClustersFrom directObject to indirectObject in u1
getCluster u2 governing NP
addSubCluster u2 to directObject in u1
```

Note that since there is no look-ahead in **IDIoT**, the cluster associated with the words 'the girl' is first assumed to be the direct object, and then, as a result of the detection of the indirect object, is corrected to being governed by feature **indirectObject**. As an alternative, it is possible that the indirect object rule becomes detected and inhibits the not-yet-detected direct object rule if the former is specified as an exception of the latter. In other words, the indirect object rule would be faster and would establish both the direct and the indirect objects by relying on the fact that the most reachable NP is the direct object, and the second most reachable one, the indirect object. A different natural language could have a different rule.

Once the syntactic cues direct- and indirect-object have been established, word expert feature(s), that is, features associated specifically with **actionGive** (much like Small's approach, 1980, 1983), can be used to modify the cluster of the verb. For example, the indirect object is the *recipient* of the action, and the direct object, the *given*. Even the **subject** feature could be renamed to **giver**. The point is that since no definitive and exhaustive list of syntactic cases exists, each verb may establish its own. These modifications can be trivially implemented with the renameFeature instruction as follows:

```
KU giveTo:
 constraint c1:
  triggers: actionGive
 expansion:
      getCluster u0 governing clause
      getCluster u1 governing actionGive in u0
      testPresenceOf indirectObject in u1
      renameFeature indirectObject to recipient in u1
      testPresenceOf subject in u0
      renameFeature subject to giver in u0
      testPresenceOf directObject in u0
      renameFeature directObject to given in u0
```

It is also possible that other rules associated with **actionGive** bypass completely the need for **directObject** and **indirectObject** to be detected and instead, directly recognize **given**, **giver**, and **recipient**.

Finally let me briefly address the handling of a simple prepositional phrase attachment in the sentence:

Example 5.2.4 ***The cat sits on the mat***

I propose the following very sketchy scenario:

1. The words 'the cat' cause an NP cluster to be built.

2. The word 'sits' leads to the detection of **actionSit** and of the subject-verb relationship.

3. The word 'on' leads to the detection of **onPreposition** which triggers the candidacy of all its possible interpretations. The feature **prepositionNP** becomes expected.

4. The words 'the mat' lead to the construction of an NP cluster. Through this process, the feature **prepositionNP** is triggered and becomes detected as it was expected. As a consequence, the cluster associated with 'the mat' is made a subcluster of the cluster constructed for 'on'. In other words, the cluster resulting from the processing of the words 'the mat' is attached to the cluster of preposition 'on'.

5. All candidate interpretations of **onPreposition** fail except **onObjectLocation**, which eventually causes (directly or through a general attachment feature) the cluster associated with 'on' to be attached as a subcluster under feature **location** in the verb cluster.

Without going into linguistic considerations that are beyond the scope of this dissertation, I remark that all the features that were presented above are still very simplistic, especially in light of hard problems such as ambiguity and ellipsis, but also due to the probable intricacy of grammar and comprehension strategies for both local and global coherence (see van Dijk and Kintsch, 1983, chapter 5). Moreover, this intricacy is independent of a user's preference for word experts (Small, 1980, 1983) or, instead, for more general rules.

In conclusion, it appears that the proposed representational methodology of **IDIoT** allows for the specification of an adequate approach to syntax. Since features can be detected at any point in time, there is no need to hypothesize an order of processing between syntax and semantics: given a particular context, some semantic rules may be faster and thus apply more readily to the context than certain syntactic rules. Indeed, the usual distinction between syntax, semantics, and pragmatics becomes superfluous in the context of reader-based comprehension: any rule is merely a piece of data processed by memory.

Chapter 6

Reference Resolution

6.1 Introduction to Reference Resolution

The problem of establishing a reference relationship between two NPs constitutes an essential facet of linguistic comprehension, especially with regards to the perception of local coherence. Several models, most of them theoretical, have been proposed, in particular for the problem of pronoun comprehension. Typically, these models address a single aspect of reference resolution. Within the framework of text comprehension, the problem is often ignored (*e.g.*, by schema-matching models such as Dyer, 1983, and Grasser and Clark, 1985) or oversimplified (*e.g.*, Norvig, 1987, reduces it to a single rather simplistic inference class).

A typical system is that of Alshawi (1987, sections 3.2 and 6.5) who proposes a general "context mechanism" that is used to find references: first, the constraints of the referent are derived and marked in a semantic spreading-activation network; second, if these constraints do not lead to a single referent (*i.e.*, a node where all the activated markers intersect), then the context mechanism is used to choose between the possible candidates. This simple and elegant mechanism can be thought of as a generalization of Grosz's notion of 'global focus' (1977).[1] In essence, the initial candidates for reference resolution are the nodes with an activation level above a certain threshold in the network. If the search in this initial set fails, then the search is widened to nodes outside this 'focus space'. More precisely, the focus space is augmented with nodes that are taken to be implicitly in focus (*e.g.*, all

[1] In Grosz's work, focus information is used for resolving definite references made in task-oriented dialogues between an expert and a novice being taught how to construct an air compressor. Focus consists of a set of nodes in the semantic network that are highlighted on the basis of relevance. For a detailed discussion of the advantages of Alshawi's model over Grosz's, see Alshawi (1987, pp.64–66).

subparts of the objects or participants that are in focus). The most serious flaw of this approach stems from this notion of implicitness that corresponds to a sort of *a priori* semantic priming of relevant features. For example, Alshawi explains (1987, p.102) that for a noun in focus, all entities below it in both the assumed specialization and 'correspondence' hierarchies are deemed implicitly in focus, and that for the pronoun 'it', all entities below the concept 'inanimate' are also considered implicit. Such a tactic is not only *ad hoc* but fundamentally irreconcilable with the philosophy of reader-based comprehension, which rejects any form of *a priori* relevance rules. Moreover, given a large knowledge base, it seems probable that an unmanageable number of features would have to be marked, especially when considering the purpose of Alshawi's correspondence hierarchy, which is to capture correspondences of the form 'role C1 of owner D1 is a role-specialization of role C2 whose owner is D2'. Consequently, the approach appears to be somewhat implausible from a cognitive viewpoint.[2]

Moreover, inference-chaining models in general (including marker-passing models such as Alshawi's) do not correlate 'focus' to the notion of reachability (and to STM constraints), and, most importantly, cannot consider clues that come after the reference in the text.

Rejecting the quest for *the* solution to reference resolution, van Dijk and Kintsch (1983, p.161) argue instead for the strategic nature of the task:

> Our view of the processes of pronoun comprehension should be seen as an integral part of our model of strategic discourse comprehension, and in particular as a part of the local coherence strategies of such a model.... However, the strategies determining the understanding of pronouns require some more specific principles. As there is much confusion in the literature about these principles, we will briefly enumerate the major ones[.]

Their proposed list comprises the following principles (*ibid.*, pp.161–164):

- Grammatical constraints: *e.g.,* number and gender agreement.
- Textual constraints: *e.g.,* referents must be 'in focus'.
- Referential constraints: pronouns do not refer to their antecedents but to individuals.
- Cognitive principles: role of situation models, capacity limitations of short-term memory.

[2] Alshawi states clearly that cognitive plausibility is not one of his goals.

These researchers also stress that the search for a referent must be performed with respect to a situation model (*ibid.*, subsection 10.1.1), that is, with respect to a schema that captures the topicality of a text. Finally, it is commonly accepted that it is most unlikely that language users will process full sentence pairs before establishing some coherence relation. Yet Kintsch and van Dijk (*ibid.*, p.164) add that "whatever the strategic operations for fast provisional interpretation, final interpretation is possible only following the interpretation of the whole clause or sentence, ... (and sometimes even of later sentences)". This observation has been generally ignored in existing computational models. The same authors also discuss several experimental results that they take to confirm the importance of expectations in reference resolution (especially for pronoun comprehension).

A complete review of the existing literature on reference resolution is beyond the scope of this dissertation (see Hirst, 1981). In the rest of the chapter, I just want to use simple examples to illustrate the adequacy of IDIoT for simple reference resolution.[3]

6.2 Reference Resolution with IDIoT

Let us first consider the problem of pronoun comprehension, for which I suggest the following general strategy in IDIoT. When a pronoun is read:

- Check its syntactic felicity at that point in the clause. In other words, verify the syntactic constraints associated with the pronoun.

- Use a findInclusiveReference instruction to establish whether there is one or several possible referents for it. A possible referent is a cluster that matches the pronoun's 'essential' features (*e.g.*, number, gender, etc.).

- If there is a single referent, replace the pronoun's cluster with the cluster of the referent. Then check for semantic disagreements (*e.g.*, the referent must be a person in the case of a personal pronoun). These disagreements capture ungrammaticalities,

[3] Complex examples such as the treatment of cataphoric or deitic pronouns are beyond the scope of most of the existing models. However, cataphoric pronouns can be tackled in IDIoT, since candidacies last a short amount of time, thus making it possible to have the referent *after* its referring NP. As for deitic pronouns, van Dijk and Kintsch (1983, p.162) claim that the treatment of this type of pronoun is quite similar to that of more usual ones. Finally, I emphasize that the features presented below are simplistic in that they do not model, in particular, the complexities resulting from a hypothesis/verification approach to resolution. More precisely, the proposed features do not model a reader who would commit to an interpretation based on expectations and then wait for confirmation or contradiction of this decision.

that is, features that do not prevent referent resolution but whose violation signals a conflict.

- If there are several possible referents, treat the pronoun as an ambiguous word (see next chapter). During the short interval of time associated with the disambiguation feature, information from subsequent words and clauses may lead to the selection of one of the possible referents.

Possible definitions for the pronoun 'he' implementing this tactic follow:

```
innate KU 'he':
% This feature recognizes the word 'he'
 associations: hePronoun
 expansion:
      getNewCluster
      addFeature 'he' to newCluster

KU hePronoun:
% This feature verifies the syntactic felicity of a pronoun at that
% point in a sentence. If the pronoun is felicitous, the feature
% marks the cluster of 'he' as a unreferenced (ie toBeMatched) pronoun.
 constraint 1:
 triggers: 'he'
 inputs:
      personalPronoun has a weight of 1
 expansion:
% Add a cluster, under feature toBeMatched, that will contain the
% features that must be matched by a referent to the pronoun. Also
% add feature unsolvedPersonalPronoun to the pronoun cluster to allow
% subsequent retrieval.
      getCluster u1 governing 'he'
      renameFeature 'he' to hePronoun in u1
      addFeature toBeMatched to u1
      getNewCluster
      addFeature noun to newCluster
      addFeature male to newCluster
      addFeature singular to newCluster
      addSubCluster newCluster to toBeMatched in u1
      addFeature unsolvedPersonalPronoun to u1
  ports:
      personalPronoun has a delay of 1 and is a supplier
      personalPronounMatch has a delay of 1 and is a customer
      personalPronounDisagreement has a delay of 1 and is a customer
```

```
KU ambiguousPersonalPronoun
% A pronoun is ambiguous iff feature personalPronounMatch fails, in
% fact saying that a unique reference for the pronoun was not found.
% In this case, further inferences will have to
% disambiguate this pronoun.
% The use of personalPronounMatch constitutes a typical use of
% exceptions in IDIoT.
 constraint c1:
  triggers: personalPronoun
  exceptions: personalPronouMatch
 ...
 expansion:
      ifConstraint c1 then getCluster u1 governing unsolvedPersonalPronoun
      removeFeature toBeMatched from u1
 ports:
      personalPronoun has a delay of 1 and is a supplier
      personalPronounMatch has a delay of 1 and is a supplier

KU personalPronounMatch:
% This feature tries to find a single referent for an unsolved
% pronoun. It could also inhibit ambiguousPersonal explicitly.
is buildable
 constraint 1:
  triggers: hePronoun
 constraint 2:
  triggers: shePronoun
 constraint 3:
  triggers: iPronoun
 constraint 4:
  triggers: himPronoun
 ...
 expansion:
      getCluster u1 governing unsolvedPersonalPronoun
      getCluster u2 governedBy toBeMatched in u1
      findInclusiveReference u3 to u2
      substitute u3 to u1
 ports:
      ambiguousPersonalPronoun  has a delay of 1 and is a customer

KU personalPronounDisagreement:
% This feature is triggered only if a personal pronoun has been solved.
% In this case, it checks whether the referent is a person. If it is not,
% a syntactic conflict is detected.
is buildable
 associations: syntacticConflict
 constraint 1:
  ordered triggers: hePronoun, personalPronounMatch
 constraint 2:
  ordered triggers: himPronoun, personalPronouMatch
 ...
 expansion:
% the most reachable NP is necessarily the referent at this point. The
% noun of this NP must be a person.
     getCluster u1 governing NP
     getCluster u1 governing noun in u1
     testAbsenceOf person in u2
     addFeature syntacticConflict in u1
 ports:
     ...
```

The purpose and details of these features are given in their comments. Moreover, these definitions do not tackle the role of expectations in reference resolution, since there is currently no agreement on the importance of syntactic and topical expectations. Also, in the case of an ambiguous pronoun, the problem is left to be solved by inference.

To illustrate this tactic, let us look at a few simple examples, starting with a passage, in which there is only one possible referent:

Example 6.2.1 *John eats. He sleeps.*

Once the resolution has terminated, the cluster associated with 'he' will be replaced for the cluster of 'John'. I remark, however, that this resolution may not be as immediate as it may appear and that, in fact, in this example, it may depend on establishing an implicit temporal inference between the two clauses (*e.g.,* John sleeps after eating). This observation suggests that, in general, a non-referential inter-clausal relationship (such as this temporal inference) may be required in order to enable or to confirm the actual resolution. In other words, the definition of **personalPronounMatch** would have to include code to check that the action of the referent comes before the action of the pronoun. The specification of such inferences is discussed in section 9.2. Compare with the passage:

Example 6.2.2 *John eats. He then sleeps.*

In this passage, the temporal inter-clausal relationship is made explicit by the connective 'then'. In this case, it appears that the resolution of 'he' to 'John' is 'stronger' (*i.e.,* less provisional, more readily confirmed) than in the preceding example, mostly because the explicit temporal link necessitates less time to establish than its implicit counterpart.

Temporal relationships are not the only inter-clausal links that may confirm or enable reference resolution. Consider the passage:

Example 6.2.3 *John eats at a restaurant. He orders caviar.*

The semantic link between the concepts 'eating at a restaurant' and 'ordering food' may help in associating 'he' with 'John'.

The definitions given above could easily be enhanced to take inter-clausal relationships into account. The importance of these inter-clausal relationships becomes more apparent in examples with multiple referents, such as this one:

Example 6.2.4 *From his hotel room, John watched the old man in the park. He envied his freedom.*

In this case, syntactic expectations (especially for agentive NPs that start clauses; see van Dijk and Kintsch, 1983, p.167) may lead the reader to disambiguate 'he' to 'John'. Similarly, topical expectations could favor 'John' as the referent. Yet, the actual resolution should only be confirmed through further inference and it is possible a reader may perceive the passage as being ambiguous.[4]

The tactic suggested above also works for relative pronouns, though these are typically simpler to disambiguate than personal pronouns since they are generally limited to prior referents. Consider, for example, a sentence in which there is only one antecedent:

Example 6.2.5 *John, who is hungry, eats.*

Again, in this case, the cluster associated with 'who' is replaced with the cluster for 'John'. As with personal pronouns, the substitution will be carried out even in the case of ungrammaticalities such as these:

Example 6.2.6 *The elephant who is hungry eats.*

Example 6.2.7 *John which I hate eats.*

The difficulty is in identifying which features are required for a resolution and which merely need to be checked afterwards by a disagreement feature. Consider the passage:

[4] Typically though, since John is the agent of 'watch' and since it is not necessarily the case that the old man also watches John, 'he' will be taken to refer to John.

Example 6.2.8 *A man with a red hat, who was jogging nearby, saw the burglar escape.*

It is clear that the referent should at least be an animate entity in order for 'red hat' to not be considered as a possible referent. In other words, 'animate' needs to be matched but 'person' may be checked only for noticing disagreement with a convention.

Let us now turn to the more general problem of definite NP resolution. **IDIoT** may use a tactic quite similar as the one suggested for pronoun comprehension: each definite NP triggers the candidacy of the buildable feature **referredNP**, whose expansion procedure tries to find a reference (*i.e.*, a complete and *unique* cluster match). If several clusters are found as possible referents, the findInclusiveReference instruction fails, the candidacy of **referredNP** aborts, and the definite NP is left ambiguous. There can be several concurrent candidacies of **referredNP**, one for each definite NP to be resolved. Moreover, when a PP (or any other syntactic unit) is attached to a definite NP, the resulting complex definite NP needs to be resolved as a whole. Thus, the attachment of an NP as a subcluster of another NP inhibits the search for a reference for either initial NPs and triggers a new candidacy of **referredNP** for the new complex NP. In other words, it is the cluster associated with the complex NP (and which includes as a subcluster, the cluster associated with the PP) that will be matched.

Here are possible definitions:

```
KU definiteNP:
% A definite NP is an NP that starts with 'the'.
 constraint 1:
  ordered triggers: theArticle, NP
 ports:
      referredNP has a delay of 1 and is a customer

KU referredNP:
 constraint 1:
  triggers: definiteNP
% The second constraint recognizes the attachment of a PP to a
% definite NP. Note that the NP that is part of the PP does not
% have to be a definite NP and is not resolved by this case.
% Other similar constraints are not shown.
 constraint 2:
  ordered triggers: definiteNP, NP, attachPPtoNP
 expansion:
      getCluster u1 governing NP
```

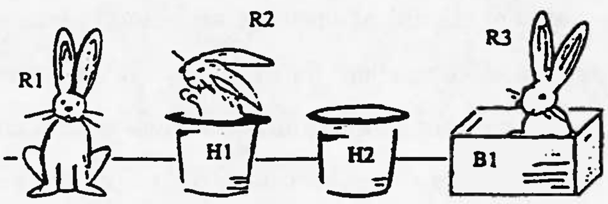


Figure 6.1: The context for Haddock's example

```
findInclusiveReference u2 to u1
substitute u2 for u1
```

(The actual reference resolution scheme of the current prototype is somewhat more complex.)

To illustrate this discussion, let us first consider Nicholas Haddock's (1987) complex NP:

Example 6.2.9 *the rabbit in the hat*

in a context in which there are three rabbits (R1, R2, and R3), two hats (H1 and H2), and one box (B1), where R1 is not in any container, R2 is in H1, and R3 in B1 (see figure 6.1).

Despite the fact there are several rabbits.in context, the NP is taken to have a unique referent. Haddock remarks that "any compositional accounts of NP semantics,... would judge [6.2.9] to be infelicitous" because they would fail to consider the referential context. Instead, he proposes an incremental interpretation stemming from a strictly word-by-word, left-to-right evaluation of the phrase (as is psychologically and intuitively well supported):

> If we assume that a hearer *incrementally* evaluates a semantic representation—after each word, say— the empty hat in the scene will never really be considered a viable candidate for the inner NP. When the word *rabbit* is reached, a hearer can collect together in his mind the set of rabbits in the context. After the preposition, this set can be refined to contain only rabbits which are *in* something and, most important, the hearer can start thinking about another set of objects, those which have rabbits in them. There is only one hat in this new set and so by the time the inner NP is processed a definite determiner sounds natural.

Haddock's approach seems to depend heavily on the unambiguous attachment of 'in' to 'the rabbit' since it is suggested that the preposition *immediately* restricts the set of rabbits to those that are *in* something. From my viewpoint, this restriction should not occur before the attachment of 'in' to 'the rabbit' is established. Therefore, to the best of my understanding, it is not clear how, in the same context, the two sentences:

Example 6.2.10 *Put the rabbit in the car.*

Example 6.2.11 *Put the rabbit in the hat.*

would be disambiguated by Haddock. In the first one, because the NP 'the rabbit in the car' has no referent in context, but also because the verb 'put' requires a direct object, the PP should not be immediately attached to the NP 'the rabbit', but to the verb 'put', and thus anyone of the three rabbits is a possible referent for the NP 'the rabbit'. In other words, the syntactic and semantic constraints of the verb must be considered by the reference resolution process. Similarly, in the second sentence, if the PP is attached to 'the rabbit', then the NP 'the rabbit in the hat' has a unique referent but the verb 'put' is left without an obligatory direct object. Alternatively, if the PP is attached to the verb, then the NP 'the rabbit' has several possible referents and is therefore ambiguous.

IDIoT's strategy does not require the complex symbolic mechanisms assumed by Haddock (*e.g.*, extension variables, combinatory grammar and its functions and arguments, etc.) and does not resolve definite NPs by set attrition. Instead, I have specified KUs that grant precedence to verb constraints over referential felicity, but otherwise use the **referredNP** feature to disambiguate NPs. Using the same context as in the preceding examples, here is a rough scenario in **IDIoT** for the sentence:

Example 6.2.12 *The rabbit in the hat is young.*

1. The definite NP 'the rabbit' triggers the candidacy of **referredNP**. Since there are multiple possible referents for this NP, this candidacy fails.

2. Another candidacy of **referredNP** is triggered by 'the hat'. Again, since there are several hats in context, this candidacy fails.

3. When, and only when, the PP 'in the hat' is attached to 'the rabbit', a new candidacy of **referredNP** corresponding to 'the rabbit in the hat' is triggered from the second constraint of **referredNP**. A unique referent is found for this complex NP and the expansion procedure of **referredNP** replaces the cluster constructed for 'the rabbit in the hat' with the cluster found as a reference. This substitution *ipso facto* resolves

both definite NPs involved in the example. From this perspective, the suggested resolution strategy is not incremental in Haddock's sense, even though it proceeds in a word-by-word left-to-right order.

4. The rest of the sentence is processed.

Here is a list of similar examples working with the current prototype of **IDIoT**:

- **Example 6.2.13** *R1 is the rabbit. R2 is the rabbit. The rabbit eats.*

  There is just one rabbit found. Much like the NP 'the mayor' corresponds to a unique role, the two first sentences are taken to refer to the same unique rabbit.

- **Example 6.2.14** *R1 is a rabbit in a hat. The rabbit in the hat eats.*

  In the second sentence, the PP is attached to the NP 'the rabbit' and then a unique referent is found.

- **Example 6.2.15** *R1 is a rabbit. R1 is in a hat. The rabbit eats.*

  A unique referent is found.

- **Example 6.2.16** *R1 is a rabbit. R2 is a rabbit. The rabbit eats.*

  Multiple referents are found, and thus there is no resolution.

- **Example 6.2.17** *R1 is a rabbit. R2 is in a box. R1 is in a hat. R2 is a rabbit. The rabbit in the box eats.*

  A unique referent is found for 'the rabbit in the box', not just for 'the rabbit'.

- **Example 6.2.18** *The young rabbit eats. The rabbit sleeps.*

  A unique referent is found. The resolution process currently ignores adjectives.

- **Example 6.2.19** *R1 is a rabbit in a hat. Put the rabbit in the hat.*

  In the second sentence, a reference is found for 'the rabbit' and the PP is attached to the verb. The PP is not attached to 'the rabbit'.

- **Example 6.2.20** *Put the rabbit.*

  A syntactic conflict is detected.

- **Example 6.2.21** *Put the rabbit in the hat.*

  Simple VP-PP attachment occurs, which blocks, in this case, the NP-PP attachement.

- **Example 6.2.22** *R1 is a rabbit. R2 is a rabbit. Put the rabbit in the box.*

  In the third sentence, no referent is found, and there is only simple VP-PP attachment.

- **Example 6.2.23** *R1 is a rabbit in a hat. R2 is a rabbit. Put the rabbit in the hat in the box.*

  The third sentence is interpreted as: 'Put in the box the rabbit that is in the hat'.

As with pronoun comprehension, inferences are extremely important for the task of definite NP reference resolution. Consider:

Example 6.2.24 *John was pale: the boy was sick.*

Example 6.2.25 *John was pale. The boy was sick and John feared he would die.*

Example 6.2.26 *John was pale. The boy was sick and John feared his son would die.*

In the first example, 'the boy' *could* be taken to refer to John, especially given the role of the colon. But this becomes less probable in the second example, in which the explicit reference to John appears to eliminate the possibility of having John refer to 'the boy'. In the third example, it is even more unlikely that John could refer to 'the boy', mostly because of the additional probable inference that 'the boy' is John's son. Such subtle inferences greatly complicate the reference resolution process.

Finally, I want to emphasize that schema-matching should not be confused with a different kind of inference that is frequent in reference resolution. In the passage:

Example 6.2.27 *John enters the park. He sits on a bench.*

I suggest there may exist a semantic link between park and bench, possibly captured in a feature **parkBench** that would be a specialization of **bench**. A reader may expect a bench after having read the word 'park'. But I propose that this expectation not be considered an occurrence of schema-matching, which I limit to the detection of 'steps' (*i.e.*, component 'subactions') of an action, as in:

Example 6.2.28 *John dines at the restaurant tonight. He orders caviar and savors it slowly. He then leaves a big tip and walks out.*

The features **orderFood**, **eatFood**, **payFood**, and **exitRestaurant** are perceived as substeps of the schematic feature **eatAtRestaurant**, that is, are organized as subclusters of the latter feature.

Chapter 7

Word Sense Disambiguation

7.1 Introduction to Lexical Disambiguation

A word (or a sequence of words) may have more than one interpretation. For example, the noun 'ball' can be interpreted as 'spherical object', 'baseball', 'testis', and 'formal dance', among other definitions. This problem of lexical disambiguation raises two difficult psycholinguistic questions:

- The problem of *lexical access*: "Do people consider (unconsciously) some or all of the possible meanings of an ambiguous word, or do context and/or expectations take them straight to the 'correct' meaning?" (Hirst, 1987, p.85).

- The problem of the *decision point*: "If more than one meaning is accessed, how and when is a choice made?" (*ibid.*).

Alternative hypotheses (Hirst, 1987, section 4.3) for the lexical access problem are the following: 1) the context may limit the process to a single access or influence disambiguation only after all possible interpretations have been accessed, or 2) interpretations are accessed in order of frequency of usage, the disambiguation process stopping as soon as an acceptable meaning is found. As for the decision-point problem, there are three possibilities: "that the choice is virtually immediate; that it does not happen until the end of the clause (or some smaller syntactic unit), with the several meanings remaining around until then; and that it happens as soon as enough information is available, whether this be immediately or later" (Hirst, 1987, p.85).

In concluding his review of a representative sample of the large body of literature on this topic, Hirst (1987, pp.94–95) writes:

> There are clearly many questions yet to be resolved in the study of human lexical access and disambiguation. However, this much seems clear: in many cases, more than one meaning of an ambiguous word is accessed. Semantic priming and frequency of a particular sense can facilitate lexical access and disambiguation, and in some cases cause one meaning to be accessed to the exclusion of others.

As we shall see subsequently, a time-constrained memory seems particularly well-suited to develop an approach to lexical disambiguation that has similar properties to Hirst's approach, which is summarized below. But first, let us consider existing research in artificial intelligence on this problem. For recent results in psycholinguistics on this topic, refer to Gorfein (1989).

Lexical disambiguation is typically reduced by schema-matching approaches to the recognition of a particular context with respect to a set of knowledge structures that specify *a priori* all recognizable contexts. For example, in BORIS (Dyer, 1983, p.181) the word 'gin' is interpreted as LIQUID if the context 'involves' the semantic primitive INGEST, and as CARD-GAME, if the context involves COMPETITIVE-ACTIVITY. Dyer (*ibid.*) adds that a word may be disambiguated in either a top-down or a bottom-up fashion. In the top-down case, disambiguation corresponds to the satisfaction of an expectation set up by the words preceding the ambiguous one. In the bottom-up case, the ambiguous word is assumed to spawn a disambiguation demon that searches the current context for a match with one of its disambiguation rules. Hirst (1983, section 4.1) explains that the choice of the knowledge structure(s) (*e.g.*, scripts, MOPS, etc.) corresponding to a given context constitutes the major difficulty of such an approach to disambiguation in that it ignores the local syntactic and semantic cues provided by nearby words.

Several models of lexical disambiguation that take into account local disambiguating cues have been proposed in artificial intelligence. For example, Hiyan Alshawi (1987) has proposed a computational model for disambiguation that is reviewed elsewhere (Corriveau, 1988). The work is not concerned with human processing at all and, in essence, describes marker-passing algorithms used to select between possible candidates for disambiguation using simple examples in a very limited domain. The *ad hoc* nature of most of these algorithms and of the flags used by the representational scheme (Alshawi, 1987, chapter 2) constitutes the major drawback of this research. Hirst (1983, section 4.1) reviews four other models. Some of these models specify more or less complex rules of disambiguation using templates (*e.g.*, Wilks's (1975) preferences). Conversely, Steven Small (1983) rejects general disambiguation rules in favor of a vast number of loosely related sets of rules associated with each individual word: the word expert approach. In his work on semantic interpretation, Hirst (1987, 1988a, 1988b) proposes the notion of *Polaroid Words*[1], which are fully integrated with his Absity semantic interpreter; upon reading an ambiguous word, Absity is given a 'fake' semantic object that acts like a self-developing Polaroid photograph of the disambiguated concept. Hirst elaborates (1987, p.97):

> By the time the sentence is complete, this photograph will be a fully developed picture of the desired semantic object. And even as the picture develops, Absity will be able to manipulate the photograph, build it into a structure, and indeed do everything with it that it could do with a fully developed photograph, except look at the final picture. Moreover, like real Polaroid photographs, these will have the property that as development takes place, the partly developed picture will be viewable and usable in its degraded form.

A Polaroid Word (hereafter PW) consists of a disambiguation procedure, one PW existing for each syntactic category. When a new PW is needed, an instance of the appropriate type is cloned and is given a packet of knowledge about the word for which it will be responsible. These packets contain only lexical knowledge in the case of nouns but include some 'world knowledge' in the case of verbs, since these "can get quite idiosyncratic about their case flags" (Hirst, 1987, p.106). Upon its instantiation and, if required, each time a new word is input, the PW attempts to narrow down its possible interpretations by looking for 'strong' paths built by the marker passing component of the model and by communicating with its 'friends'.[2] The PW eliminates any of its meanings that are incompatible with those of its friends. As for the paths supplied by the marker-passing component, their 'strength' is evaluated with respect to 'magic numbers' set with respect to the two following heuristics (*ibid.*, subsection 5.6.3):

- The shorter the path, the stronger the path.

[1] *Polaroid* is a trademark of the Polaroid Corporation.

[2] "Verbs are friends with the prepositions and nouns they dominate; prepositions are friends with the nouns of their prepositional phrase and with other prepositions; and noun modifiers are friends with the noun they modify. In addition, if a prepositional phrase is a candidate for attachment to a noun phrase, then the preposition is a friend of the head noun of the NP to which it may be attached.... The intent of friendship constraints is to restrict the amount of searching for information that a PW has to do; the constraints reflect the intuition that a word has only a very limited sphere of influence with regard to selectional restrictions and the like" (*ibid.*, subsection 5.3.2). Also, the rules of friendship can become somewhat complex when dealing with bound constituents (*ibid.*, subsection 5.3.5).

- The more arcs that leave a node, the weaker the connections through that node.

Summarizing, Hirst (*ibid.*, p.111) writes:

> Polaroid Words with marker passing are not a replacement for inference and pragmatics in word sense and case disambiguation; rather, they serve to reduce substantially the number of times that these must be employed.

The current implementation of Polaroid Words does not have such an inference or pragmatics system available to it and does not use syntactic cues nor the 'global context' (for which there is no representation). Also, the treatment of figurative language remains problematic in that such usage typically violates the rules of disambiguation postulated for literal English (*ibid.*, section 5.4).

From my viewpoint, the major drawback of Hirst's work is that the complex algorithms and data structures used for Polaroid Words may be acceptable in the framework of semantic interpretation (an approach I abandon, as explained earlier) but are somewhat irreconcilable with the restrictions and considerations imposed by cognitive modeling.[3] And, as is generally the case with marker-passing systems, Polaroid Words rely heavily on 'magic numbers' (*ibid.*, subsection 5.6.3).

7.2 Lexical Disambiguation with IDIoT

As mentioned in chapter 2, and in accordance with reader-based hermeneutics, this research does not focus on some abstract language as incompletely specified by a dictionary, nor on some 'literal English' resulting from an artificial distinction, but rather on the *idiolect* of a specific user. In other words, the notion of a 'universal lexicon', that is, a correct and exhaustive list of all possible meanings for any given word in a language, is abandoned. It is the user who must decide on the 'boundaries' (or conceptual distance) between concepts.[4]

Similarly, it seems that existing algorithms, which statically capture rules for disambiguation, are limited to a literal interpretation of words and can never take into account the actual run-time context of a reading. Small (1983) views language as being too irregular for generalizations above the lexical level. Instead, he suggests the coding of an individual

[3] For a discussion of the psychological reality and non-reality of PWs, see Hirst, 1987, section 5.6.

[4] Do pigs fly? Yes they do, when on board an airplane. How close is this meaning of the verb 'fly' to the ones in the sentences "Time flies" and "Birds fly"? Concepts are often thought of as well-defined objects when, in fact, they encapsulate 'shades' of meaning. Conceptualization has its limits (Sowa, 1984, chapter 7).

and large disambiguation procedure for each word, an approach that requires a lot of time, reflecting the idiosyncratic nature and long apprenticeship of linguistic communication; this displeases those researchers who search for regularities in language. Though Small's strategy is undeniably inconvenient from an engineering point of view, I feel very close to it for, in the spirit of reader-based comprehension, it is based on the acceptance of human idiosyncrasies rather than on the quest for the correct set of rules and algorithms. The fundamental difference between **IDIoT** and Small's work is that I attempt to 'de-proceduralize', that is, to avoid the use of *a priori* static algorithms for the specification of his word experts by having the user express them by means of the representational scheme presented in chapter 3. Furthermore, as will be shown below, there is nothing to prevent the user of **IDIoT** specifying in the KB the rules of disambiguation that, as an individual, he assumes to be correct.

In the same vein, the notions of polysemy, homonymy, and categorial ambiguity are not considered useful in this discussion, for they introduce a semantic distinction that is irrelevant to a trivial algorithm. It is left to the user of **IDIoT** to specify features corresponding to rules that would or would not take this distinction into account.

I repeat that, as with grammar, the specification of the 'correct' set of rules of disambiguation is taken to be highly problematic, for it would ignore both the presence of a context that cannot be specified *a priori*, and the effects of real-time processing, which stipulates that certain rules are more retrievable than others. In this chapter, the problem of disambiguation is addressed from a quantitative viewpoint and, as in the rest of this thesis, no claim is made on how concepts should be codified.

The proposed model of memory is particularly well-suited to implement a disambiguation strategy similar in spirit to Hirst's (1987) Polaroid Words. In **IDIoT**, words are disambiguated over a period of time, that is, the set of the concepts that are candidate interpretations for a given word w may shrink over time. I develop below **IDIoT**'s approach to disambiguation, not with respect to a particular corpus of examples, for this typically leads to an algorithm fitted to the data, but at a more abstract level, that is, in terms of w, an unspecified ambiguous word in some language.

If the word w is ambiguous, then, at the time of its detection, all possible interpretations of w are notified and can become candidates for detection. This is easily realized by having these interpretations specified as the candidates (see chapter 3) of w. Recall that each candidate has a certain retrievability at that given point in time, and that communication is taken to be asynchronous. Therefore, all possible interpretations of w become candidates, *but* these candidacies do not necessarily start at the same time. And, in the case where one of the possible interpretations of w is already being expected, if the signal received from w triggers it, then it immediately becomes detected. Also, the expectation mechanism in **IDIoT** can be used to model the phenomenon of semantic priming, as illustrated in the next section.

If candidate x has candidate y specified as an exception of its triggered constraint, then the candidacy of x necessarily takes its maximum time-span in order for y to be given the opportunity to become detected and possibly inhibit x. Since exceptions are not automatically reciprocal, y may not have x as an exception and therefore may become detected at as soon as it can be. Thus, the user may specify through the exceptions of the possible interpretations of w a disambiguation strategy that favors some candidates over others by having the former being immediately detectable while the latter must take the maximal time allocated to a candidacy. The ability for one KU to dynamically modify the retrievability of another, an enhancement discussed in chapter 10, would make this strategy more dynamic.

Once triggered, a candidate has a fixed amount of time to become detected, as explained in chapter 3. Disambiguation may also occur through the process of reference resolution. Detection by referencing is possible only if the ambiguous word is explicitly repeated after having been disambiguated in the nearby context, as in the following example:

Example 7.2.1 *John poured the gin but Mary refused: she had always hated gin.*

The first occurrence of 'gin' is disambiguated to the feature **alcoholicBeverageGin** by constructing a path to the verb 'pour' (as in the example of section 3.7). If STM capacity allows for a context that spans at least a few words, then a reference is quickly found for the second occurrence of 'gin', which is thus also disambiguated to the feature **alcoholicBeverageGin**. This referencing ability, which does not appear to exist in the previously overviewed models of lexical disambiguation, seems to be a plausible psychological approach, one where a repetition sufficiently 'close' (with respect to time and STM capacity) to the original disambiguation, is not treated as an ambiguity but as a readily detectable reference. Hirst has pointed out that using a content word with two different senses in the same sentence usually leads to a garden path.

Since candidacies may span the processing of a few words (or even clauses) after the input of *w*, they are not limited to considering the context existing at their start, but in fact can take into account the context as it is modified over their time-span. In other words, both the evidence existing at the start of a candidacy and the information established from subsequent words can affect disambiguation.

In **IDIoT**, the act of disambiguation *per se* proceeds from the execution of the expansion procedure of a possible interpretation that becomes detected. This procedure first accesses the cluster with the original ambiguous word *w*, and then proceeds to modify this cluster. At an abstract level, if *x* is a possible interpretation of *w* that becomes detected, then the expansion procedure of *x* could either rename the feature **w** to **x**, or simply add **x** to the cluster of *w*.
Subsequent instructions of the expansion procedure of *x* could typically add the features that define *x*, as well as specialize some of the features associated with *w*. For example, disambiguating the word 'gin' to the alcoholic beverage could replace the innate feature **'gin'** with feature **alcoholicBeverageGin** in the cluster created by **'gin'** and add some information about **alcoholicBeverageGin**. Since all qualitative data is user-specified, the word 'gin' may be disambiguated to a more general feature **alcoholicBeverage** that would govern another cluster capturing the kind of alcoholic beverage, as in the following expansion procedure:

```
getCluster u1 governing 'gin'
addFeature alcoholicBeverage in u1
getNewCluster u2
addFeature kindOf to u2
addSubCluster u2 to u1
getNewCluster u3
addFeature gin to u3
addSubCluster u3 to u2
```

In summary, the disambiguation step *per se* simply consists of accessing the cluster of *w* and modifying it to reflect the disambiguation of *w* to *x*; the specificity of the resulting cluster depends entirely on a specific user's KB.

Similarly, it is left to the user of **IDIoT** to decide on a feature-by-feature basis which candidates 'know' about and can affect the candidacy of the others. More precisely, if candidate x is an exception of the triggered constraint of candidate y, or if x is explicitly sent an inhibition signal by the output strategy of the triggered constraint of y, then the detection of y will inhibit the candidacy of x.

Possible disambiguation scenarios include the following:

- **Winner-take-all**: a candidate becomes detected and causes all other candidacies to fail.
- **Insufficient conclusive evidence**: all candidacies fail.
- **Mutually exclusive detections**: several candidacies can succeed, reflecting an ambiguity.
- **Successive disambiguations**: several candidacies successively succeed.

If, for example, **IDIoT**'s user systematically specifies all possible interpretations of w as mutual exceptions, then the first scenario will be followed: the first possible interpretation x to become detected inhibits all other candidacies. This case is possible as long as x has the exceptions and output strategy of its triggered constraint send inhibition signals to all the other candidates. Recall, however, that, as always, it is up to the user to judge the merits of such an approach. The second scenario is always possible: none of the candidates manages to become detected for lack of evidence. In this situation, w is left ambiguous. The third scenario is the trickiest and corresponds to the less frequent situation where the ambiguity may be consciously perceived, possibly by having several possible interpretations come to mind. I suspect that an adequate solution to the latter case involves nothing less than a theory of self-awareness, which is absent from **IDIoT**. The fourth scenario is always possible, although it can be made somewhat infrequent through the judicious use of exceptions which, I repeat, have the advantage of forcing candidacies to take their maximal time-span, and thus of minimizing the risk of a conflict between an interpretation that would ignore other candidates and become quickly detected, and a less retrievable one that would nonetheless become detected in light of its existing context.

IDIoT's approach to disambiguation is similar to Hirst's (1987) Polaroid Words in that:

1. A semantic object, namely the cluster constructed by w, is initially associated with the ambiguous word w.

2. This semantic object is just another element of STM, and thus is fully integrated with the rest of the context and can be manipulated by other KUs.

3. This semantic object ends up containing a developed picture, that is, the disambiguated feature, if any.

However, in **IDIoT**, there is no need for *a priori* rules constraining how much of the context can be considered.

In conclusion, I remark that **IDIoT**'s approach to disambiguation does not fit any of the hypotheses mentioned earlier for the problems of lexical access and decision point, but rather synthesizes all of them through the *modus operandi* of time-constrained memory. First, for lexical access:

- A single access is possible according to a winner-take-all scenario (*e.g.*, resulting from a triggered expectation or a quickly retrieved candidate which inhibits all others).

- An all-readings approach is possible if all candidacies overlap in time and yet lead to a single winner.

- Frequency of usage is accounted for in the retrievability coefficient of the possible interpretations of *w* on which the whole disambiguation process ultimately depends.

As for the decision-point problem, there is no need to postulate either the highly problematic 'immediate decision' approach or any arbitrary semantic criterion, since the actual decision point may vary from feature to feature and according to an unpredictable context. The purely quantitative detection mechanism of the time-constrained memory handles the problem by simply limiting the time that may be spent disambiguating.

7.3 Examples of Lexical Disambiguation

Let us start by considering the problem of semantic priming. I emphasize again that, in view of recent criticism by Dennis Norris (1986), no claim is made with regards to the psychological reality of this phenomenon. Let us consider the sentence:

Example 7.3.1 *John plays gin.*

A feature x is said to prime a feature y if, upon the detection of x, an expectation signal is sent from x to y, which lowers the retrievability coefficient of y and places it in expectation mode.

Let us make the following assumptions regarding the KB:

- Innate feature **'plays'** eventually triggers **actionPlay**.
- Feature **actionPlay** primes **cardGame**.[5]
- Feature **cardGame** is an association of **cardGameGin**; **game** is an association of **cardGame**.
- Feature **'gin'** has **alcoholicBeverageGin** and **cardGameGin** as candidates.
- Both **alcoholicBeverageGin** and **cardGameGin** are triggered by **'gin'**.

Upon the detection of **actionPlay** (from the word 'plays'), **cardGame** is primed and becomes expected. After the word 'gin' is input and detected, **alcoholicBeverageGin** and **cardGameGin** are triggered and become candidates. Feature **cardGameGin** sends a submission signal to its association **cardGame**. Upon receiving this signal, **cardGame** is triggered, has its expectation satisfied, and thus, since this a submission, sends a confirmation signal to its submitter **cardGameGin**. After receiving this confirmation signal, **cardGameGin** becomes detected and disambiguates, by means of its expansion procedure, the cluster governing **'gin'**. The detection of **cardGameGin** leads to the detection of **cardGame**, which, in turn, leads to the detection of the more abstract feature **game**.

If **cardGameGin** were directly primed by **actionPlay**, no submissions would be necessary as it would become immediately detected by being triggered by **'gin'** while being expected. And if **actionPlay** primed **game** instead of **cardGame**, the chain of submissions would go from **cardGameGin** to **cardGame** to **game**, reflecting the cost of a more abstract (*i.e.*, less specific) KB.

Let us consider the same sentence, but without any priming in the KB. Let us adopt, for example, the following KB assumptions:

- Innate feature **'plays'** eventually triggers **actionPlay**.

[5] Thus, **actionPlay** would also probably prime all other types of games, which would result in a KB with a high degree of specificity. The KB could be even more specific if **actionPlay** directly primed **cardGameGin** and all other known games, as may be the case for a child.

- Feature **cardGame** is an association of **cardGameGin**. **game** is an association of **cardGame**.

- Both **alcoholicBeverageGin** and **cardGameGin** are possible interpretations of **'gin'**.

- Feature **actionPlayAGame** and **actionPlayAMusicalInstrument** are possible interpretations of **actionPlay**.

- Feature **actionPlayAGame** has a constraint that consists of the two triggers **actionPlay** and **game**.

- Feature **game** is a relay with, among others, **cardGame** as a supplier and **actionPlayAGame** as a customer.

The word 'plays' eventually leads to the detection of **actionPlay**, which triggers the candidacy of **actionPlayAGame** and **actionPlayAMusicalInstrument**. It is assumed the latter will not have enough evidence to become detected and eventually have its candidacy expire. As for **actionPlayAGame**, it is still missing the trigger **game** after receiving the signal from **actionPlay**. From the input of the word 'gin', **'gin'** is recognized, which eventually leads to the candidacies of **alcoholicBeverageGin** and of **cardGameGin**. It is also assumed that **alcoholicBeverageGin** will not have enough evidence to become detected and eventually have its candidacy expire. Upon receiving the signal from **'gin'**, **cardGameGin** becomes a candidate and a submission signal is sent from **cardGameGin** to **cardGame** to **game** to the customers of **game**. These customers include **actionPlayAGame**, which is satisfied by this submission signal and sends a confirmation signal back to **game**, eventually leading to the detection of this chain of features.

I have used a relay for **game** to suggest that the 'highest' (*i.e.*, most general) feature of a particular generalization hierarchy of objects (implemented by the associations of the concerned features) can often be too general to require an expansion procedure and therefore can be specified as a relay. It is left to the user to avoid a deadlock where **game** would need **actionPlayAGame** to become detected and *vice versa*. Experimentation with the current prototype has shown that such deadlocks can be prevented through the judicious use of relays for such 'abstract' features.

A strictly conceptual approach to disambiguation has its disadvantages. In the last hypothesized KB, **actionPlayAGame** becomes detected merely from the co-occurrence of **game** and **actionPlay** and, therefore, could erroneously specialize **actionPlay** to **actionPlayAMusicalInstrument** in the following example:

Example 7.3.2 *The baby played with the guitar. (Hirst, 1987, p.78)*

or to **actionPlayAGame** in the sentence:

Example 7.3.3 *John plays with his gin.*[6]

A simple rule to prevent this would be to insist that the verb 'play' have a direct object if it is to be specialized to **actionPlayAGame** or **actionPlayAMusicalInstrument**. Such a syntactic check can be enforced by having these features as buildable features with the following check at the start of their respective expansion procedure:

```
getCluster u1 governing actionPlay
testPresenceOf directObject in u1
...
```

In this case, the co-occurrence of the required triggers would not be sufficient and the existing context would be checked by means of a testPresenceOf instruction. Disambiguation of **actionPlay** to one of its possible interpretations would depend on this check.

But this check is not sufficient for a sentence like:

Example 7.3.4 *John plays Hamlet with a gin in his hands.*

The co-occurrence of the required triggers and the presence of a direct object still lead to an erroneous interpretation. A more demanding rule may insist that, for example, **actionPlay** be disambiguated to **actionPlayAGame** only if the game in question is the direct object of the verb 'play'. In other words, **actionPlayAGame**, for example, could require another check in its expansion procedure to verify that the cluster associated with the direct object, let's call it *DO*, denotes (*i.e.,* has some features that denote) a game. Such an approach is problematic in that *DO* will not denote a game until **'gin'** has been disambiguated, but

[6] For example, John makes noises while drinking his gin, or John toys with his glass.

'gin' being disambiguated to **cardGameGin** necessitates a confirmation from feature **actionPlayAGame**, which has its syntactic check on the direct object fail because **'gin'** is not yet detected. In other words, the detection of **cardGameGin** would need a confirmation from **actionPlayAGame**, whose syntactic check would require that **cardGameGin** have been detected.

This chicken-and-egg problem is typical of lexical disambiguation, and syntactic rules are unhelpful, in this specific case, for they can only establish that the word 'gin', not the concept **cardGameGin**, is the direct object of the word 'play'. Clearly, the specification of such a detection cycle must be avoided. The key to the problem is deciding which of the words 'play' and 'gin' is to be disambiguated first. I present below a possible solution in which 'play' is disambiguated only once 'gin' has been specialized to the card game, and leave the reciprocal alternative as an exercise for the reader. The 'trick' is to have the syntactic check(s) and chain of features that will allow the detection of **cardGameGin** involve not **actionPlayAGame** (which will become detected only once **cardGameGin** has been disambiguated) but **actionPlay**. In other words, since it is assumed 'play' is disambiguated *after* 'gin', the disambiguation process relies on the co-occurrence of the concept **actionPlay** with some 'playable' direct object.

Here is a possible KB that summarizes this discussion and combines both syntactic and conceptual considerations:

- Innate feature **'plays'** eventually triggers **actionPlay**.
- Innate feature **'gin'** eventually leads to the detection of the relay **noun** and of a feature **directObject**, which governs the cluster associated with **'gin'**
- Feature **cardGame** is an association of **cardGameGin**; **game** is an association of **cardGame**.
- Features **alcoholicBeverageGin** and **cardGameGin** are possible interpretations of **'gin'**.
- Features **actionPlayAGame** and **actionPlayAMusicalInstrument** are possible interpretations of **actionPlay**.
- Feature **actionPlayAGame** has a constraint that consists of the two triggers **actionPlay** and **game**.

- Feature **game** is a relay with, among others, **cardGame** as a supplier and **actionPlay** as a customer. This modification circumvents the problematic cycle.

- Feature **objectGin** has a constraint with the triggers **'gin'** and **directObject**, and has the following expansion procedure:

```
getCluster u1 governing directObject
testPresenceOf 'gin' in u1
```

- Feature **cardGameGin** has a constraint with only the triggers **'gin'** and **objectGin**.

The word 'plays' eventually leads to the detection of **actionPlay**, which triggers the candidacy of **actionPlayAGame** and **actionPlayAMusicalInstrument**. It is assumed the latter will not have enough evidence to become detected and eventually have its candidacy expire. As for **actionPlayAGame**, it is still missing the trigger **game** after receiving the signal from **actionPlay**. From the input of the word 'gin', **'gin'** is recognized and notifies **objectGin**. Also, the constructed cluster of **gin** becomes the direct object of 'play'. The feature **directObject** becomes detected in this process of establishing the direct object and sends its presence signal to **objectGin**, which, in turn, becomes detected and notifies **cardGameGin**. Feature **alcoholicBeverageGin** becomes a candidate from the presence of **'gin'**, but again it is assumed that **alcoholicBeverageGin** will not have enough evidence to become detected and eventually have its candidacy expire.

When **cardGameGin** receives the signal from **objectGin**, a submission signal is sent from **cardGameGin** to **cardGame** to **game** to the customers of **game**. These customers include **actionPlay**, which is satisfied by this submission and sends a confirmation signal back to **game**, eventually leading to the detection of this chain of features. Then, and only then, does the detection of **game** lead to the detection of **actionPlayAGame**.

I remark that the proposed solution illustrates a case- or role-based approach to conceptualization that is often taken to be pervasive to human cognition (see Sowa, 1984): the specification of a particular concept involves the role of the concept. Finally, I claim that a strategy for disambiguation that would flag the word 'gin', and only this word, as having multiple possible interpretations would miss the fact that the word 'play' itself requires a lot of knowledge to interpret (as suggested by Small's (1983) approach). For example, the following sentence is deemed to be nonsensical by some of my informants while quite understandable by others:

Example 7.3.5 *The game plays well.*

If we admit the inchoative 'play' ought to be disambiguated to **actionPlayAGame**, then we realize that another constraint of **game** or of **actionPlay** must account for 'game' being the subject (or agent) of the verb 'play', which precisely reinforces the two previous observations, namely:

- Importance of case- or role-based conceptualization.
- Frequent need for individual word experts.

Let us now consider the case of an homonymous word such as 'submarine' in the sentence:

Example 7.3.6 *The sailor ate the submarine. (Hirst, 1987, p.88)*

This sentence can trigger a garden-path effect if the idiosyncratic processing of a comprehender has the word 'sailor' strongly prime the concept of the undersea ship rather than the concept of the sandwich. Hirst (personal communication) has tried similar sentences on informants and reports that most found the sentences 'funny' and could not 'see' the literal interpretation.

Let us assume that the verb 'eat' has no figurative interpretation.[7] Paths from the possible interpretations of 'submarine' (*e.g.*, sandwich or undersea ship) are buildable to both 'sailor'[8] and 'ate', yet only the latter seems relevant to disambiguation. An explanation of this intuition may involve the retrievability of the concerned features: if the sandwich interpretation is very unfamiliar and very long to retrieve, or if the undersea ship meaning is extremely retrievable, then it is possible that disambiguation will fail (leading to the garden path effect).

If we assume equal retrievability, then it is very possible that the path between 'ate' and **submarineSandwich** (*e.g.*, submarine–sandwich–food–eat) is the shortest to establish (with respect to processing time) and then inhibits the undersea ship candidacy. However,

[7] This is not always the case, *e.g.*, "to eat one's words", "the sea eats ships and submarines", etc.

[8] A sailor is a type of person. A person eats food. A sandwich is a type of food.

if we postulate that the undersea ship interpretation is unacceptable simply because a undersea ship cannot be the direct object of the literal verb 'eat', an *inhibition* path[9] between **submarineShip** and **eat** could be specified: **submarineShip** would become a candidate triggered by **objectSubmarine**[10] and inhibited by a path such as submarineShip–metalConstruction–nonEdibleObject–actionEat. Again, however, we would have to assume this path is necessarily built before a path is constructed between 'sailor' and 'submarine'. Furthermore, as usual, the presented chains of features are simplistic and the user must specify a real word expert in order to account for the disambiguation of 'submarine' in the following passage:

Example 7.3.7 *Each Easter, it seems one can find more and more bizarre objects fashioned in chocolate. This year, the Pentagon released miniatures of its arsenal in milk chocolate. Yesterday, my son ate a submarine but he still prefers an F-14.*

In this paragraph, the attribute of 'submarine' that made it inedible is replaced (in the corresponding cluster through a series of complex inferences) by chocolate, a kind of food. Therefore, an inhibition path between 'submarine' and 'eat' becomes unlikely. In other words, the previous rule suggested for disambiguation is too simple.

It is important that, through the interval of time of a candidacy, information that follows the ambiguous word also be taken into account, as in the sentence:

Example 7.3.8 *John eats a submarine made out of chocolate.*

This example also illustrates the possible dangers of inhibition paths (which are not currently available in **IDIoT**): if, after reading 'submarine', the candidacy of **submarineShip** is immediately inhibited as a result of the inhibition path between **submarineShip** and **actionEat**, then subsequent information that would block this path (*i.e.*, prevent its creation) would be ignored. The point of these examples is that the disambiguation process ultimately depends on the idiosyncratic inferences of the comprehender.

Let us continue with an example of categorial ambiguity, as the word 'sink' in the sentence:

[9] In the case of an inhibition path, proper confirmation to the original submitter would inhibit the latter, rather than have it become detected.

[10] A feature, similar to **objectGin**, which is used to make sure that 'submarine' is the direct object of **actionEat**.

Example 7.3.9 *John emptied the sink of its water.*

The word 'sink' can, in theory, be a noun or a verb. However, in the example, syntactic cues restrict the choice to the noun. More precisely, I propose that the word 'sink' would trigger the candidates **sinkNoun** and **sinkVerb**, that would respectively submit the features **noun** and **verb**. A confirmation path would be obtained between **noun** and the existing syntactic context—the detection of the article 'the' would either expect or confirm the feature **noun**—eventually leading to the detection of **sinkNoun**. The latter feature would, in turn, notify its possible interpretations, which could include **plumberSink**, **viceSink**, **heatSink**, etc. Similarly, if **sinkVerb** were detected, it would notify its possible interpretations.

It appears that if a rigid grammar, that is, one that systematically rejects ungrammatical sentences, is specified, then the resolution of categorial ambiguity relies mostly on syntactic cues (*e.g.*, which sequences of parts-of-speech are allowed). But, as suggested in section 5.1, the mind typically possesses an enormous ability to handle 'ungrammaticality' that can lead to more complex cases of categorial ambiguity, as with the word 'bark' in the sentence:

Example 7.3.10 *The dog's bark woke me up.*

Two other interesting examples handled by the current prototype of **IDIoT**, in which several linguistic problems are combined into a single sentence, are:

Example 7.3.11 *The men man the submarine.*

Example 7.3.12 *The sailors man the submarine.*

In extreme cases, categorial ambiguity can create a 'garden path' effect (Marcus, 1980) as in the sentence:

Example 7.3.13 *The prime number few.*

Interestingly, this sentence often seems to present much less difficulty for a native French reader than for a native English reader. An explanation may reside in the historical development of these two languages: like Latin, French is an analytic (as opposed to a synthetic) language where syntactic cues appear to be much more rigid and 'verb-oriented' than the English ones, tending to prevent the French reader from making decisions until the verb has been found, and thus avoiding the 'trap' of interpreting 'prime number' as the mathematical idiom. The point to be repeated is that both ungrammaticality and garden-path sentences suggest complex syntactic cues that allow for some flexibility in parsing but also can 'jump to conclusions' leading the reader 'down the garden path'.

Let us now consider a sentence in which several words are ambiguous, as in the sentence:

Example 7.3.14 *Nadia's plane taxied to the terminal. (Hirst, 1987, p.126)*

The word 'plane' has several possible interpretations (*e.g.*, plane tree, tool for smoothing wood, geometric plane, aircraft) as a noun, and as a verb (*e.g.*, to make smooth, to remove by planing). 'Taxi' can be a noun or a verb. And the word 'terminal' can, at least, be associated with **computerTerminal** and **airportTerminal.** Since 'plane' has a categorial ambiguity, upon the recognition of **'plane'**, I suggest that the features **planeNoun** and **planeVerb** become candidates, eventually leading, through syntactic cues, to the detection of the former. Once **planeNoun** becomes detected, it notifies all its possible interpretations. A given KB may cause the **aircraftPlane** to become detected as a result of a direct path between **actionTaxi** and **aircraftPlane.** In the case of a less fluent reader who ignores the fact that 'to taxi' specifically applies to airplanes, the path between **actionTaxi** and **aircraftPlane** will simply be longer (*e.g.*, actionTaxi–actionVehicleMovement–vehicle–aircraftPlane).

The recognition of the preposition 'to' causes its possible interpretations to become candidates. However, a path between 'to' and **actionTaxi** is probably built (*e.g.*, actionTaxi–actionMove–movement–destination–destinationTo), leading to the disambiguation of **'to'** to **destinationTo.**

The recognition of **'terminal'** leads to the candidacy of **computerTerminal** and **airportTerminal.** If 'plane' has been disambiguated to **aircraftPlane** from 'taxied', it is probable that either an expectation, or a path between **airportTerminal** and **actionTaxi** or **aircraftPlane** will lead to the disambiguation of 'terminal' to **airportTerminal.** However, if the knowledge of the reader had not allowed for the disambiguation of 'plane', the disambiguation of 'terminal' would probably fail, since both of its possible interpretations can act as destinations of a movement. Consider, for example, the sentence:

Example 7.3.15 *John walked to his terminal and compiled his software.*

Again, the word 'terminal' as the destination of a movement does not provide useful information (since a passenger of an aircraft may also walk to the terminal). If, in the previous example, the disambiguation of 'terminal' could rely on 'plane' or 'taxied', it seems that in this case 'terminal' will be specialized to **computerTerminal** using a path that would probably involve **software**. Such a scenario again emphasizes the importance of having candidacies that span a few words after their triggering input.

Another important observation with respect to disambiguation is that, in **IDIoT**, inferences are completely integrated with the quantitative detection mechanism described in chapter 3. Consider, for example, the sentence:

Example 7.3.16 *The view from the window would be improved by a plant. (Hirst, 1987, p.126)*

An inference is assumed to be required in order to disambiguate the word 'plant' to **vegetalPlant** instead of **industrialPlant**. Let me briefly digress to stress that **vegetalPlant** is not *necessarily* the adequate interpretation. For example, a rich industrialist may prefer seeing a new factory he owns rather than some stupid vegetable that presents no interest whatsoever to his pragmatic sense of esthetics! A new building may also represent an improvement over some ugly sight. End of digression. If we assume that the sentence *in vacuo* favors the **vegetalPlant** interpretation, then the word 'plant' will be disambiguated through features and paths that capture the complex hypothesized inference.[11]

Referential inferences are also relevant to disambiguation, as suggested earlier. Consider the sentence:

Example 7.3.17 *The astronomer married the star.*

Let us assume that some movie actress has been recently referenced. In this case, a cluster *c* denoting this specific actress has been constructed and includes the feature **actorStar**. Upon the recognition of the word 'star' of the example, **celestialBodyStar** and **actorStar** both become candidates. A reference is found for **actorStar** in the context if *c* is reachable

[11] For example:

1. The words 'the view from the window' lead to the detection of **windowView** and of its association **roomLocation** and of its association **interiorLocation**.
2. The word 'improved' leads to the detection of **actionImprove**.
3. The fact that **interiorLocation** is the object of **actionImprove** causes the detection of **interiorDecorationImprovement**.
4. The path (vegetalPlant, windowOrnamanent, ornament, additionOfRoomOrnament, interiorDecorationImprovement) is found leading to the confirmation and detection of **vegetalPlant**.

This proposal emphasizes the importance of generalizations using associations but is still simplistic, especially in that it does not account for the subtleties stemming from the use of a conditional.

and **actorStar** becomes detected. This strategy respects the intuition that a reference should eliminate the need for the disambiguation process.

Finally, I remark that for sentences (from Hirst, 1987, p.126) such as:

Example 7.3.18 *I want to eliminate some moles.*

Example 7.3.19 *Ross was escorted from the bar to the dock.*

if there is not enough information to disambiguate, then the candidacies of the possible interpretations will simply expire, without making the reader aware of these multiple potential interpretations, since there is currently no model of awareness in **IDIoT**.

7.4 On Idioms and Figurative Language

Some NLP models need some counterintuitive *a priori* warning in order to process so-called figurative (*i.e.*, non-literal) language. It is as though understanding required yet some more specialized algorithms to tackle this pervasive (Lakoff and Johnson, 1980) aspect of language that is generally ignored. From my viewpoint, there can be no *a priori* distinction between a literal and a figurative interpretation, for this would imply some sort of improbable prescience: each reader typically produces a private interpretation without knowing in advance what a text is about. In this section, I very briefly investigate how the problem of figurative language can be approached with **IDIoT**.

Let us first consider the simplest form of idioms. (For a more detailed discussion of idioms, see Miezitis, 1988.) It is my opinion that lexical and structural ambiguities can often be eliminated through an efficient processing of idioms, which, I believe, always consist in a pattern whose global interpretation as a whole is faster than, and does not necessarily proceed from, the interpretation of its individual components. For example, conventionalized (verb–preposition) patterns could be specified as idioms (*e.g.*, 'give up', 'fall in love', etc.), *de facto* minimizing the number of possible interpretations for both verb and preposition. Clearly, this approach favors the word expert strategy that was previously advocated, an exhaustive list of idioms being specified for each verb and preposition.

In **IDIoT**, an idiom may be specified as a feature **i** with the components of the idiom's pattern as a set of ordered triggers. Once triggered, **i** has its constraint(s) check for the semantic felicity of the idiom's interpretation in the existing context. More precisely, the key feature(s) of the interpretation are specified as inputs in these constraints. When an idiom has an interpretation that cannot be derived from the individual interpretation of its pattern's components, success or failure in being detected involves only the constraints of the idiom. Conversely, an idiom that can also be interpreted literally will typically be detected if one of its constraints is satisfied before the literal interpretation can be assembled. Idioms very often correspond to lexical patterns. The case where such a lexical idiom can also be interpreted literally (*i.e.*, have an interpretation built from its components) is treated as a lexical ambiguity with the only difference that, if we assume equal retrievability of all relevant features, the idiom will be a candidate sooner, as it is directly triggered by a pattern of words that requires less time to detect than that required for assembling them into an interpretation. Let me clarify this discussion by considering the idiom 'to kick the bucket', for which I propose the following definition:

```
KU kickTheBucket:
 constraint 1:
  ordered triggers: rootKick, 'the', 'bucket'
  inputs:
      actionDie has a weight of 1
  exceptions:
      actionKick has a weight of 1
      bucket has a weight of 1
expansion:
      getCluster u1 governing rootKick
      renameFeature rootKick to actionDie in u1
      addFeature kickTheBucket to u1
      getCluster u1 governing 'the'
      removeCluster u1
      getCluster u1 governing 'bucket'
      removeCluster u1
```

This feature is essentially triggered by **rootKick** (*i.e.*, the morphological stem of the verb 'kick') followed by the words 'the bucket'. The input **actionDie** is submitted unless it has already been detected, in which case the constraint would be immediately satisfied. Once **actionDie** is confirmed by the context, the idiom becomes detected, provided neither of its exceptions has been detected. The expansion procedure simply takes note of the idiomatic form of **actionDie** and removes the clusters associated with the words 'the bucket'. Finally, the feature **kickTheBucket** must also be specified as an exception of **actionKick** and of **bucket** in order to inhibit the literal interpretation. Consider, for example, the sentences:

Example 7.4.1 *John kicked the bucket; the milk spilled.*

Example 7.4.2 *John kicked the bucket. His widow celebrated.*

For both sentences, the features **actionKick, bucket,** and **kickTheBucket** will become candidates. In the first sentence, **actionDie** will not be confirmed, and thus the candidacy of the idiom will expire without success. In fact, an inference between 'bucket' and 'spilled' could probably be established, causing the detection of **bucket**, and thus the inhibition of **kickTheBucket**. In the second sentence, a confirmation path having been built between **actionDie** and 'widow', **kickTheBucket** becomes detected.

Idioms can be viewed as conventionalized figurative language, their idiomatic interpretation not requiring any special treatment: in the process of becoming conventionalized by a reader, the initial inferences that are made to understand them are lost and only the interpretation as a whole remains. In other words, language is not static, it has a history that is generally unknown and irrelevant to the comprehender, who is taught an idiomatic interpretation as a single lexical item. If we think of figurative language usage as a spectrum, idioms are at one end: they have been conventionalized to the point where they constitute lexical items. In the middle of the spectrum we find the conventionalized metaphors (*e.g.,* "argument is war", "instrument as companion", etc., see Lakoff and Johnson, 1980) that pervade our use of language. Finally, at the other end of the spectrum, reside today's novel metaphors (which may become tomorrow's idioms). It is precisely the novelty of a usage, with respect to an individual's idiolect, that forces the comprehender to establish a new inference path (which may be missed or require more time to construct due to its complexity).

Consider, for example, the passage:

Example 7.4.3 *As a writer, John had been humiliated; his pen breathed revenge (adapted from Hirst, 1987, p.115)*

The context favors the disambiguation of 'pen' as 'writing implement' rather than as 'female swan' (via a path between 'writer' and 'pen'). The comprehender must also:

- Recognize the conventionalized metaphor "instrument as result" which takes a pen as meaning a writer's work.

- Set the interpretation of 'to breathe' to 'to mean', 'to express'.
- Make the link between revenge and humiliation and possibly infer a causal scenario such as "the humiliated seeks revenge".

Each of these three steps may seem quite straightforward (if not quite literal) for an experienced speaker of English.[12] Yet they may be quite complex for a beginner who must either be taught the interpretation or must invest considerable time and effort in order to construct an interpretation by herself. The immediacy of an interpretation also varies with how well the words fit a comprehender's conventions, as illustrated in the following sentences:

Example 7.4.4 *His pen perspires revenge.*

Example 7.4.5 *His pen spits revenge.*

Example 7.4.6 *His pen produces revenge.*

Example 7.4.7 *His pen generates revenge.*

Example 7.4.8 *His pen inhales revenge.*

Example 7.4.9 *His pen exhales revenge.*

Example 7.4.10 *His pen smells revenge.*

Example 7.4.11 *His pen satisfies his revenge.*

And, as always, in the worst case, the passage will be misunderstood or not understood.

The possibility for misunderstanding or not understanding increases as we move on the axis of figurative usages. Consider the first few lines of John Keats's "Ode on a Grecian Urn":[13]

> Thou still unravish'd bride of quietness,
> Thou foster-child of silence and slow time,
> Sylvan historian, who canst thus express
> A flowery tale more sweetly than our rhyme.

[12] As a matter of fact, the two first steps are so conventional that they can be found in a dictionary or a thesaurus.

[13] In: Carl Bain, Jerome Beaty, and J. Paul Hunter (eds.) (1977) *The Norton Introduction to Literature*, Second edition, W.W. Norton & Company, New York.

The interpretation will not only completely escape a 'down-to-earth' kind of comprehender (*e.g.*, how can an urn, which is inanimate object, be an unravished bride, a foster-child, an historian?) but will also remain somewhat incomplete in the case of a reader who is not familiar with the Rousseauist movement.

The point is that figurative language can only be defined with respect to one's idiolect, one's experiences and knowledge, one's postulates and conventions: one man's evidence can be another man's mystery. In other words, figurative language ought not to be an *a priori* notion but an *a posteriori* classification of a usage with respect to a particular individual. From this standpoint, there is no need to hypothesize separate interpretation mechanisms: a novel usage merely requires more complex, possibly less immediate inferences. And, in the end, the ascription of meaning also depends on idiosyncratic acceptability, fear of losing face, authority, power (Peckham, 1979): an individual may feel compelled to produce an interpretation for a surrealist poem, painting, or sculpture that he considers meaningless but that is regarded as a masterpiece by his teacher.

Chapter 8

Structural Disambiguation

8.1 Introduction to Structural Disambiguation

The problem of structural disambiguation stems from the fact that a sentence may have several parses. For a detailed discussion of this problem, see chapter 6 of Hirst (1987). Here is his table (*ibid.*, p.135) summarizing attachment ambiguities in English:

- *PP attachment—to noun or verb?*
  Ross insisted on phoning the man with the limp.
  Ross insisted on washing the dog with pet shampoo.
- *PP attachment—to which noun?*
  the door near the stairs with the "Members Only" sign
- *Relative clause attachment—to which noun?*
  the door near the stairs that had the "Members Only" sign
- *PP attachment—to which verb or adjectival phrase?*
  He seemed nice to her.
- *PP attachment—to which verb?*
  Ross said that Nadia had taken the cleaning out yesterday.
- *Adverb attachment—to verb or sentence?*
  Happily, Nadia cleaned up the mess Ross had left.
- *Participle attachment—to surface subject or sentence*
  Considering his situation likely to go from bad to worse, he decided to offer his resignation.
  Considering the deficiencies of his education, his career has been extraordinary.

Following is his table (*ibid.*, p.149) of analytic ambiguities in English:

- *Relative clause or complement?*
  The tourists objected to the guide that they couldn't hear.
  The tourists signaled to the guide that they couldn't hear.
- *Particle detection*
  A good pharmacist dispenses with accuracy.

- ***Prepositional phrase or adjectival phrase?***
  I want the music box on the table.
- ***Present participle or adjective?***
  Ross and Nadia are singing madrigals.
  Pens and pencils are writing implements.
- ***Present participle or noun?***
  We discussed running.
- ***Where does an NP end?***
  Nadia gave the cat food.
  The prime number few.
- ***Reduced relative clause or VP?***
  The horse raced past the barn fell.
- ***Determining noun group structure***
  airport long term car park courtesy vehicle pickup point
- ***What is the subject of the supplementive?***
  He drove the car home undismayed.
  He brought the car back undamaged.
- ***Supplementive, restrictive relative, or verb complement?***
  The manager approached the boy smoking a cigar.
  The manager caught the boy smoking a cigar.
- ***Cleft or not?***
  It frightened the child that Ross wanted to visit the lab.
- ***Question or command?***
  Have the crystals dissolved?
  Have the crystals dissolved.
- ***How is the predicate formed?***
  Ross is eager to please.
  Ross is ideal to please.
  Ross is easy to please.
  Ross is certain to please.

These tables are not claimed to be exhaustive, and structural disambiguation also involves other problems such as gap finding and filling (see *ibid.*, section 6.2.2) as in these sentences:

Example 8.1.1 *Those are the boys that the police debated about fighting.*

Example 8.1.2 *Mary is the student whom the teacher wanted to talk to the principal.*

Structural ambiguities can involve categorial ambiguities and can lead to a garden-path phenomenon, as one will have probably experienced by reading some of the previous examples. Also, it is generally accepted that a sentence that presents such an ambiguity has a preferred parse. However, Lenhart Schubert (1986) remarks that a discussion of preferences often degrades to a battle of "partisan informants", who often do not even agree on whether a sentence is 'confusing' or not. A review of current theories for this problem can be found in Hirst (1987, section 6.3) who concludes:

> There is at present no agreement on any general principles that can be used for disambiguation. It seems clear, however, that knowledge from several different sources is used.

For example, most of the studies on attachment decisions originate in the principles of Right Association and of Minimal Attachment, which are purely syntactic (McRoy, 1988, p.6):

> The principle of Right Association states that optimally, terminal symbols will be attached to the lowest non-terminal node that is on the right-most branch of the current structure; that is, they will be grouped with the terminal symbols immediately to their left.... Minimal Attachment,... requires that optimally a terminal symbol is to be attached into a parse tree with the fewest possible number of new non-terminal nodes linking it with the nodes already in the tree.

The principle of Lexical Preference, which, in essence, states that lexical verbs and other lexical items may prefer one pattern of complementation to another, also plays an important role in recent discussions on attachment priorities. Against the syntactic trend, Yorick Wilks (1975, *et al.* 1985) argues for a more semantic version of lexical preferences, in which preferences correspond to selectional restrictions. Both Hirst (1983, 1987) and Schubert (1986) offer models that synthesize all these factors. Let us very briefly overview each of these two models.

Hirst suggests the use of a Semantic Enquiry Desk (SED), which is systematically consulted by his Paragram parser for assistance with prepositional phrase attachment and gap finding in relative clauses. The SED requires (Hirst, 1987, p.167):

- An annotation on each verb sense as to which of its cases are 'expected' (COMPULSORY, PREFERRED, or UNPREFERRED).
- A method for deciding on the relative plausibility of PP attachment.
- A method for determining the presuppositions that would be engendered by a particular PP attachment, and for testing whether they are satisfied or not.
- A method for resolving the issue when the strategies give contradictory recommendations.

For the second requirement, Hirst observes (1987, p.168) that:

> In the most general case, deciding whether something is plausible is extremely difficult.... However, there are two easy methods of testing plausibility that we can use that, though non-definitive, will suffice in many cases. The first of these,... is the slot

> restriction predicates.... While satisfying the predicates does not guarantee plausibility, failing the predicates indicates almost certain implausibility. The second method is what we shall call the *EXEMPLAR PRINCIPLE*: an object or action should be considered plausible if the knowledge base contains an instance of such an object or action, or an instance of something similar.

For the third requirement, the SED relies on a referential heuristic that is very close to the second method of plausibility testing. Hirst develops specific decision algorithms for the fourth requirement (*ibid.*, pp.173–174). These *a priori* rules ignore inferences, context, and pragmatics. A full discussion of the SED and of its results can be found in Hirst (1987, chapter 7).

Schubert's approach to PP attachment is a lot more sketchy, involves numerically weighted preferences, and also allows for trade-offs among syntactic and semantic/pragmatic preferences. The model relies on the following six principles (1986, pp.601–602):

1. **A graded distance effect:** Immediate constituents of a phrase prefer to be close to the *head lexeme* of the phrase. The effect is mediated by an 'expectation potential' that decreases with distance from the head lexeme and increases with constituent size; as a result, larger constituents admit larger displacements from the head lexeme.

2. **A rule of habituation effect:** There is an inhibitory potential or 'cost' associated with each phrase structure rule (including lexical rules), leading to a preference for low-cost rules over high-cost rules.

3. **Inhibition by errors:** 'Mild errors' such as concord errors contribute inhibitory potentials to the phrases in which they occur.

4. **Salience in context:** The potential of a word sense or phrase is high to the extent that the denotation of that word sense or phrase is salient in the current context.

5. **Familiarity of logical-form pattern:** The potential of a phrase is high to the extent that its logical translation instantiates a familiar pattern of function-argument combination.

6. **Conformity with scripts/frames:** The potential of a phrase is high to the extent that it describes a familiar kind of object or situation (such as might be specified in a script or frame).

The first two principles are taken to capture syntactic preferences, and the others, semantic and pragmatic effects. In particular, the fourth principle "is intended to allow for semantic priming by spreading activation" (*ibid.*, p.602).

Closest to the *modus operandi* of time-constrained memory, Susan McRoy (1988) proposes a psychologically plausible model of parsing that provides a good account of memory constraints, sentence complexity, structural preferences, and verb-frame preferences. The model, which is based on the Sausage Machine Model (Frazier and Fodor, 1978; Fodor and Frazier, 1980) employs a principled theory of grammar (such as the theory of Government and Binding, see Chomsky (1982)), concurrently processes syntax and semantics, and uses estimated timing information to resolve conflicting preferences. Both McRoy's work and mine are based on the fundamental idea of viewing comprehension as a race process. A most important difference between our two models resides in the fact that **IDIoT** has a strong connectionist flavor to it whereas McRoy's parser is algorithmic and symbolic.[1] Also, the role of time in her work is limited to *attachment hypothesizers* (McRoy, 1988, pp.40–42) used to choose between conflicting preferences. As argued throughout my thesis, time plays a much more pervasive role in time-constrained memory and, ultimately, linguistic comprehension. And finally, McRoy's research focuses on the parsing of single sentences, not inference.

8.2 Structural Disambiguation with IDIoT

As with the rest of part 2, my goal in this section is not to propose a solution to the problem of structural disambiguation, but rather, to suggest how time-constrained memory emphasizes some of the quantitative aspects of this problem that are generally ignored when the 'correct' interpretation is always sought. Let me start with two preliminary comments on the existing research.

First, I agree with Hirst and McRoy in acknowledging that structural disambiguation (hereafter, SD) involves several sources of knowledge. Hence I discard both purely syntactic or semantic approaches to SD, and focus instead on the syntheses proposed by Hirst and Schubert. As a matter of fact, I strongly disagree with solutions that assume and depend on

[1] The symbolic rules and mechanisms required by McRoy to implement her grammar are replaced in IDIoT by the *modus operandi* of Knowledge Units in time-constrained memory and by the restricted language of the expansion procedures of these KUs.

a syntax-first strategy and/or a particular parser with specific data structures, operations, and limitations (*e.g.,* Marcus, 1980). Consequently, in the framework of a trivial algorithm where all qualitative rules (including syntax, semantics, pragmatics, inferences, etc.) are uniformly processed rather than artificially separated, I see no reason to discuss purely theoretical parse trees and complex hypothetical parsers.

Second, when considering the questionnaire used by Schubert (1986, p.605) to quiz his informants[2], and the debate surrounding "partisan informants", I am not only reminded of Spiro's (1980) and Dillon's (1980) warnings on the artificialness of such experiments, but also disappointed by the fact that the idiosyncratic nature of comprehension, which should be acknowledged from the term 'preference' and from the debate surrounding Schubert's results[3], is completely ignored.

The fundamental hypothesis with respect to IDIoT's treatment of SD is that there is no need to postulate separate mechanisms and algorithms but, on the contrary, that disambiguation, if it occurs, will proceed from the same processes as those assumed for lexical disambiguation. In other words, I claim that the mechanisms of time-constrained memory suffice for the specification of the rules used by an individual to solve structural ambiguities. (As mentioned earlier, there is no need to hypothesize the set attrition mechanism proposed by Haddock (1987).) And again, I repeat that my goal is not to specify a correct set of such rules but to illustrate IDIoT's ability to capture these rules.

Time-constrained memory seems well-suited to implement a model of SD which amalgamates some facets of the proposals of Hirst and Schubert while respecting the philosophy of Small's (1980, 1983) word experts. I develop this model below, briefly focusing on the differents problems of SD from an abstract viewpoint, rather than with respect to specific examples.

Let us consider, in some detail, the problem of attachment. Put in a simplistic way, a word x needs to be attached to some other element of the context, which will be called the 'hook' for x. Prepositional phrase attachment consists in hooking the preposition that

[2] The questionnaire asked one to read the test sentence "at *normal* speed" and to "immediately" answer the question that followed it "as *honestly* as one could" (my italics). The answers one had to choose from requested that one decide whether one became *self-conscious* of an ambiguity, or of a need to reanalyze, or of a plausibility judgment, or unconsciously obtained the correct interpretation.

[3] With each of his test sentences, Schubert gives a fraction that shows the proportion of subjects who reported initially arriving at an anomalous reading. These fractions vary from 24 percent to 90 percent and are typically rejected by Wilks *et al.* (1985).

starts the PP; relative clause attachment, the relative pronoun that starts the clause; adverb attachment, an adverb by itself; and participle attachment, the participle that starts the subordinate proposition. From this perspective, the problem of attachment can be seen as consisting in the lexical disambiguation of the word x to be hooked.

Given that lexical disambiguation in **IDIoT** is a time-constrained process, it is possible that x be left ambiguous at the end of its candidacy, an alternative seldom implemented in the existing models. Consider, for example:

Example 8.2.1 *John insisted on drying the dog with a scarf. (adapted from Hirst, 1987)*

Since 'drying with a scarf' and 'dog with a scarf'[4] are both more or less plausible, the preposition 'with' could be left ambiguous unless contextual evidence caused one of the two possible interpretations (namely **withInstrument** and **withAttribute**) to become detected. Similarly, consider:

Example 8.2.2 *He seemed nice to her. (Hirst, 1987, p.135)*

This sentence is ambiguous unless the context is taken into account. In **IDIoT**, 'taking into account the context' simply means having the possibility of constructing one or more confirmation paths to the reachable clusters of short-term memory. Also affects the use of rules: certain rules of disambiguation may be less retrievable than others at a given point in time.

If we assume that x can be hooked, then we must consider the nature of this hook. Recall that from my standpoint, syntax is mostly irrelevant by itself in that it merely constitutes a mechanism to speed up the recognition of certain semantic cues. Also, it is generally accepted that the principles of Right Association and Minimal Attachment do not offer, by themselves, an adequate solution to the problem of attachment: semantics are required for attachment (Hirst, 1987, section 6.3). Consequently, as with lexical disambiguation, I suggest abandoning the quest for principles, which always seem to admit counterexamples, in favor of the 'word expert' approach advocated by Small (1983). This strategy completely agrees with viewing attachment as consisting in the lexical disambiguation of the hook. From this standpoint, principles are not banished but merely reify a disambiguation tactic that some, if not most, word experts will adopt. However, each individual word expert

[4] For example, Scottish terriers are often depicted with scarfs.

need not plead allegiance to these principles: individual lexical idiosyncrasies, in fact, each rule their domain, the monarchy of principles simply being a convenient illusion that is acknowledged or ignored. Moreover, IDIoT's underlying time-constrained memory allows for a more or less direct implementation of Schubert's (1986) first two principles, which he assumes to capture syntactic preferences for attachment decisions:

- In IDIoT the idea that the immediate constituents of a phrase prefer to be 'close' to the head lexeme of this phrase can be correlated to the fact that features that will be attached to a cluster need to be detected 'close' (in time) to the feature that causes the construction of the cluster. Otherwise, it will become unreachable and the attachment will not be possible.

- The notion of an 'expectation potential' that decreases with distance from the head lexeme can be directly captured using IDIoT's expectation mechanism, where the chances of detecting an expectation decrease with respect to time, and thus, as more inputs are processed (the number of processed inputs defining Schubert's notion of 'distance').

- The combination of IDIoT's reachability and expectation mechanisms can capture Schubert's idea of a constituent's trade-off between wanting to be close to its head lexeme (in order to insure the reachability of the latter), and becoming expected (due to the detection of the head lexeme), which allows this constituent (because of the possible greater time-span of an expectation over a 'normal candidacy') to be at a greater 'distance' (in terms of the time between the two detections) of its head lexeme.

- The retrievability coefficient of each KU, and thus of each user-specified rule and preference, handles Schubert's rule habituation effect: the less frequent (*i.e.*, the more 'expensive' to retrieve) a rule or preference is, the less chance it has of being considered.

I observe that, if Schubert's first two principles do capture syntactic preferences[5], then such preferences can be explained, in IDIoT, from a strictly quantitative point of view.

[5] Schubert's approach is still too sketchy to explain why the following examples (from Hirst, 1987, section 7.2.1), the first two of which are ungrammatical, are intelligible, implying the reader's ability to make the appropriate attachment(s):

- Nadia for his birthday gave her secretary a gyroscope.
- Nadia gave her secretary for his birthday a gyroscope.

Moreover, the relevant quantitative mechanisms are not fitted to the problem at hand but, quite on the contrary, are designed totally independently from any qualitative consideration, reinforcing my claim that the strictly quantitative *modus operandi* of a time-constrained memory pervades linguistic comprehension.

Schubert's three remaining semantic principles of salience in context, familiarity, and conformity (with schemas) are also accounted for in **IDIoT**:

- Confirmation paths can only be built from a candidate to the reachable context.
- Familiarity is captured by the retrievability coefficients of the relevant features.
- Conformity with schemas can be handled with expectations (Dyer, 1983).

The point is that the quantitative facets of time-constrained memory can basically account for all facets of at least one model for attachment decisions, while still viewing such a decision as consisting in the lexical disambiguation of the word to be hooked. However, Schubert's model being sketchy, it does not address some of the semantic tasks that Hirst assumes to be necessary for attachment decisions, namely, plausibility judgments, presupposition identification and testing, and trade-off rules for when the strategies give contradictory recommendations. How do these tasks fit **IDIoT**'s approach to attachment?

Plausibility judgments are reduced in Hirst's work to two separate processes, namely, selectional restrictions and the examplar principle (see previous section). Selectional restrictions form an important facet of lexical disambiguation and have been discussed when considering the sentence "The sailor ate the submarine" in the previous chapter. With respect to attachment, these restrictions would simply allow for the confirmation of one of the possible interpretation of the hook. For example, consider:

Example 8.2.3 *Ross loves the girl with a passion. (Hirst, 1987, section 7.2.4)*

I propose the following scenario: From the co-occurrence of **actionLove** and **withPreposition** (respectively triggered from the words 'love' and 'with'), the feature **lovesWith** becomes a candidate. Its detection requires that the cluster corresponding to the noun

- The gyroscope for Nadia's secretary gave him great pleasure.
- Nadia gave the secretary on the second floor a gyroscope.

phrase associated with the preposition (in this case, 'a passion') include the feature **mannerQuality**. This tactic implements the selectional restriction that states that 'with' can be attached to 'love' "only as the MANNER case, but requires the filler to be a **manner-quality**" (*ibid.*). The detection of **withPreposition** also triggers the possible semantic interpretations of 'with' including **withAttribute** whose detection attaches the PP to the preceding noun for which it constitutes an attribute. A cluster with feature **mannerQuality** is then constructed as a result from the processing of the words 'a passion'. At this point in time, the buildable feature **loveWith** has its restriction satisfied and becomes detected. The candidacies of the other interpretations of 'with' could be inhibited or left to expire on their own.

If the sentence (adapted from Hirst, 1987, section 7.2.4) were:

Example 8.2.4 *Ross loves the girl with the brown eyes.*

then **loveWith** would never become detected. Instead, 'eyes' being an attribute of 'girl', the feature **withAttribute** would eventually be detected, without any plausibility judgment based on what I would call the 'referential felicity' of the string 'the brown eyes' with respect to 'the girl'. Hirst instead suggests the examplar principle to make a plausibility judgment on the NP-PP attachment: if a reference to 'girl with the brown eyes' (or something 'similar') is found in the KB, then plausibility is granted and attachment can proceed. From my viewpoint, plausibility judgments are so problematic[6] that attachment should not rely on them. For example, if the sentence is:

Example 8.2.5 *Ross loves the girl with purple polka dot eyes.*

then the relative implausibility[7] of such eyes should not prevent attachment to 'girl'. Similarly, I minimize the importance of testing for presupposition satisfaction for attachment decisions. Consider, for example:

Example 8.2.6 *John loves the ocelot with thε bluε chipmunk. (adapted from Hirst, 1987, section 7.2.5)*

Let me quote at length Hirst's (*ibid.*) description of the relevant presuppositions:

[6] Implausibility is seldom perceived by the reader in the fictive worlds invented by authors (Graesser and Clark, 1985, subsection 1.3.5).

[7] In its unending quest for a better life in the civilized world, technology now brings us contact lenses to modify the colour of our iris.

> First, a definite NP presupposes that the thing it describes exists and that it is available in the focus or knowledge base for felicitous (unique) reference; an indefinite NP presupposes only the plausibility of what it describes. Thus, '*a* blue chipmunk', presupposes only that the concept of a **blue chipmunk** is plausible; '*the* blue chipmunk further presupposes that there is exactly one blue chipmunk available for ready reference. Second, the attachment of a PP to an NP results in new presuppositions for the new NP thus created, but cancels the uniqueness aspect of the referential presuppositions of both constituent NPs. Thus, 'the ocelot with the blue chipmunk' presupposes that there is just one such ocelot available for reference (and that such a thing is plausible); the plausibility and existence of an **ocelot** and a **blue chipmunk** continue to be presupposed, but their uniqueness is no longer required. Third, the attachment of a PP to a VP creates no new presuppositions but rather always indicates new (unpresupposed) information.

Haddock's (1987) approach to such complex definite NPs has already been criticized in section 6.2. In **IDIoT**, I propose that the preposition 'with' is attached to the NP 'the ocelot' without any plausibility judgment or presupposition testing on 'the blue chipmunk', 'the ocelot', or 'the ocelot with the blue chipmunk'. I do, however, recognize the importance of 'referential felicity' and of Hirst's exemplar principle, a simple version of which can be implemented using the **referredNP** feature discussed in section 6.2. Let me elaborate by sketching a possible scenario for the last example. For clarity and simplicity, let us assume that the context 'talks' of a nursery in a zoo where feline cubs are placed with smaller animals to play with, but that there is no explicit reference to either an ocelot or a chipmunk. Let us also postulate that attachment will result from the detection of the feature **withColocation** (which captures a co-location relation between its two arguments).

After reading the string 'the ocelot', a cluster x is constructed for it and the buildable feature **referredNP** becomes a candidate (see section 6.2). Since it is assumed there is no earlier reference to an ocelot, this candidacy will fail. Conversely, if such a reference existed and was reachable, then **referredNP** would immediately become detected and x would be replaced by this reference. And, if there was a reference to 'an ocelot with a blue chipmunk', the matching process would succeed because **referredNP** uses a findInclusiveReference instruction. Again x would be replaced by this reference.

The recognition of the word 'with' leads to the candidacy of all possible interpretations of 'with' (including **withAttribute** and **withColocation**). A cluster y is then constructed for the NP 'the blue chipmunk' and a new **referredNP** candidacy starts for 'the blue chipmunk'. Again, since it is assumed there is no earlier reference, this candidacy will fail. If a specific and reachable reference to a blue chipmunk existed, then y would be replaced by it. The context having suggested the co-location of animals, the feature **withColocation** is eventually detected, and other relevant candidacies expire or are inhibited. The execution of this feature's expansion procedure causes y to become a subcluster in x, governed by feature **colocation**. To reflect that the attachment of y to x forms a new noun phrase, a new candidacy of **referredNP** would be triggered as a result of the detection of **withColocation**. This third consecutive candidacy of **referredNP** would succeed if a reachable cluster describing an ocelot co-located with a blue chipmunk could be found. However, since it is assumed no such reference exists, this candidacy will also eventually fail.

In summary, the attachment is realized through the disambiguation of 'with', without considering plausibility or presuppositions. However, the referential felicity of each relevant presupposition is tested by means of the repeated candidacies of **referredNP**. This strategy, which restricts the search for a reference to reachable clusters during a short interval of time, seems psychologically more plausible then a search over a complete KB (as seems to be required by the exemplar principle). Finally, in the case of indefinite noun phrases, there is no referential check, as **referredNP** is simply not triggered. This approach is still simplistic and the user may wish to specify features that would account for the subtleties of *de re versus de dicto* readings.

Last, Hirst develops (1987, subsection 7.2.6) an algorithm that defines priorities for making attachment decisions when the results of verb expectation and presupposition and plausibility testing do not agree. This algorithm is limited to clauses with one VP and one NP, and does allow some counterexamples (*ibid.*, p.191). Similar priorities could be implemented in **IDIoT** by either making less probable attachment rules less retrievable (an idea similar to Schubert's principles) and/or by specifying triggers for the less probable rules that would insure that more probable alternatives have been considered. However, since **IDIoT** favors an approach to attachment that does not rely on plausibility or presupposition, but rather on the lexical disambiguation of the hook, it seems that such a complex scheme is not necessary: the retrievability of the rules used to disambiguate the hook, and the expectations and context that affect such a disambiguation should generally suffice. In other words, in **IDIoT**, the complexity of attachment decisions is shifted from *a priori* procedures to user-specified rules. Furthermore, an attachment decision does not necessarily have to be made. Consider, for example, the following sentence, which is generally taken to be ambiguous:

Example 8.2.7 *Nadia saw the man in the park with the telescope. (Hirst, 1987, p.175)*

At least, the 'with the telescope' PP should probably be left ambiguous, that is, either unattached or attached to 'saw', 'man', and 'park'. In other words, in the case of an ambiguity, no actual disambiguation should occur unless desired (and thus, set up by preferences) by the user. And, as with lexical ambiguities, capturing the perception of the different alternatives of such an ambiguity would probably require a theory of awareness.

Finally, the proposed approach to the problem of attachment favors a word-expert strategy, disambiguation features being associated with hooks and with any juxtaposition features involving the hooks (*e.g.*, feature **loveWith**). Such an approach carries over to other problems of structural disambiguation. Consider, for example, the last problem of Hirst's list of analytic ambiguities (see previous section) and its associated examples, which are repeated below (Hirst, 1987, p.149):

1. Ross is eager to please.
2. Ross is ideal to please.
3. Ross is easy to please.
4. Ross is certain to please.

In each sentence, disambiguation can only proceed from the specific characteristics of the attribute and its possible path to both 'Ross' and 'to please'. Similarly, the problem of deciding between a relative and a complement in the sentences (*ibid.*):

1. The tourists objected to the guide that they couldn't hear.
2. The tourists signaled to the guide that they couldn't hear.

respectively involves the word experts associated with 'objected' and 'signaled'. In fact, when studying Hirst's list of analytic ambiguities in English, one can identify the following factors that must be considered:

- Word experts (*e.g.*, *relative clause or complement?*).
- Syntactic restrictions (*e.g.*, a verb is required in the *particle detection* example).
- Context (*e.g.*, *prepositional phrase or adjectival phrase?*) .
- Inference (*e.g.*, *what is the subject of the supplementive?*).

- Importance of time-constrained memory (*e.g.*, garden path effect in *reduced relative clause or VP?* and *determining noun group structure*).

- Idioms (*e.g.*, "lay down the law").

My claim is that all these factors can be accounted for in **IDIoT**'s approach to comprehension, as suggested in this second part of the dissertation.

8.3 Examples of Structural Disambiguation with IDIoT

Let us focus on the problem of PP attachment. Consider the sentence:

Example 8.3.1 *The women discussed the tigers on the beach. (Hirst, 1987, p.175)*

From my viewpoint this sentence can be ambiguous and the disambiguation of 'on' will depend on the retrievability of the different possible interpretations and on the context. The same holds for:

Example 8.3.2 *The women discussed the dogs on the beach. (ibid.)*

for which Hirst prefers the NP-attachment. To illustrate the importance of context and inference, consider the following passages:

- Feature **NP-attached**: After supper, the two couples moved to the living room. The men talked about baseball. The women discussed the dogs (or tigers) on the beach that were causing so much trouble with the tourists that summer.[8]

- Feature **VP-attached**: The women discussed the dogs (or tigers) on the beach, while sunbathing, and then went to the cottage to talk about the children.[9]

Let us develop a possible simple scenario for each of these examples. (A more complex attachment scheme, which recognizes forced attachment to an NP or to a verb, has been developed with the current prototype of **IDIoT**.) For both passages, the ambiguous preposition 'on' is assumed to lead to the candidacy of its possible interpretations including

[8] If this scene takes place in some exotic paradise or fictive world, the 'tigers on the beach' is as likely and as much a problem for tourists as 'dogs on the beach'.

[9] If these women are (or are close to) breeders, zoo keepers, etc. or love (or possess, etc.) exotic pets, then tigers are as likely a conversation topic as 'dogs'.

onActionLocation and **onObjectLocation**, which respectively capture the verb attachment (*i.e.*, action performed in a specific location) and the noun attachment (*i.e.*, object existing in specific location). Here are possible definitions for these features.

For feature **onActionLocation**:

```
constraint 1:
 triggers: actionVerb, onPreposition
 exceptions: <all the other interpretations of 'on'>

 expansion:
      getCluster u1 governing actionVerb
      testAbsenceOf location in u1
      getCluster u2 governing onPreposition
      getCluster u3 governedBy nounPhraseOfPP in u2
      testPresenceOf location in u3
      addFeature location in u1
      addSubCluster u3 to location in u1
      removeCluster u2
```

Unordered triggers are used so that attachment to the verb is not limited to the case where the PP follows the verb. The expansion procedure first checks that the verb does not have a specified location. If it does, the procedure fails. Otherwise, the noun phrase of the PP is checked for the feature **location**. If it has it, then attachment to the verb occurs. The cluster associated with the preposition in itself is eliminated since its semantic function has been captured by the feature under which the attachment is placed (in this case, **location**). This strategy can be applied to any preposition.

For feature **onObjectLocation**, the definition is quite similar:

```
constraint 1:
 ordered triggers: NP, onPreposition
 exceptions: <all the other interpretations of 'on'>

 expansion:
      getCluster u1 governing NP
      testPresenceOf object in u1
      getCluster u2 governing onPreposition
      getCluster u3 governedBy nounPhraseOfPP in u2
```

```
testPresenceOf location in u3
addFeature location to u1
addSubCluster u3 to location in u1
removeCluster u2
```

For this feature, triggers are ordered, since attachment requires that the PP follow the NP to which it is attached. Also, in the expansion procedure, u1 will be bound to the most reachable NP, that is, the NP that immediately precedes the PP.

A scenario for the first passage follows:

1. In the first sentence, the word expert for **actionMove** concludes that the subject (*i.e.*, the two couples) of **actionMove** end up in in the living room. More precisely, the detection of **actionMoveToLocation** causes the cluster associated with the subject to add the feature **location** that is set up to govern the cluster associated with the living room.

2. In the second sentence, an inference path is built between 'the men' and the previously processed 'the two couples' to recognize the implicit reference.

3. From the detection of this implicit reference, another rule deduces that the two arguments of this reference are still in the same 'time frame' [10].

4. Since the two arguments of the reference are in the same time frame, yet another rule infers that the action performed by the men (*i.e.*, **actionTalk**) occurs in the current location of the men, that is, through the reference, the living room. The feature **location** is added to the cluster of **actionTalk** and is made to govern the cluster associated with the living room.

5. In the third sentence, the same rules lead the reader to infer that the action performed by the women (*i.e.*, **actionDiscuss**) occurs in the current location of the women, that is, the living room.

6. The words 'the dogs' or 'the tigers' lead to the construction of an NP cluster that becomes the direct object of **actionDiscuss**. The word expert associated with 'discuss'

[10] This inference involves the 'tense rules' of English, which allow a reader to perceive a change or an absence of change in narrative time. The notion of 'tense rules' and of 'time frame', which are taken to be crucial to text interpretation, are discussed in section 9.2.

then substitutes feature **topic** for feature **directObject** in the **actionDiscuss** cluster. More specifically, the co-occurrence of **actionDiscuss** with a direct object leads to the detection of the feature **actionDiscussWhat** which makes the direct object cluster associated with **actionDiscuss** its 'topic' by making the direct object cluster become a subcluster of **actionDiscuss** governed by feature **topic**. The processing of 'the dogs' or 'the tigers' also leads to the candidacy of **referredNP** through the detection of the **NP** feature. For simplicity, this candidacy is assumed to fail.

7. The word 'on' leads to the detection of **onPreposition**, which triggers the candidacy of all its possible interpretations. The feature **prepositionNP** becomes expected.

8. The words 'the beach' lead to the construction of an NP cluster. Through this process, the feature **prepositionNP** is triggered and becomes detected since it was expected. As a consequence, feature **NP** is replaced by feature **nounPhraseOfPP** in the cluster constructed from the processing of 'the beach', and this cluster is made a subcluster of the cluster constructed for 'on' under feature **NP-PP**. In other words, the cluster resulting from the processing of the words 'the beach' is attached to the preposition 'on'.

9. Since the cluster of **actionDiscuss** does have feature **location**, the testAbsenceOf instruction of its expansion procedure systematically fails. Consequently, the candidacy of **onActionLocation** never succeeds. All other possible interpretations of 'on', except **onObjectLocation**, also fail from a lack of evidence.

10. The candidate **onObjectLocation** becomes detected if it finds the feature **location** in the NP cluster it governs. In the example, the processing of the word 'beach' does lead to the construction of a cluster that includes feature **location** in order to capture the fact that a beach is a location. Thus, **onObjectLocation** has enough information to become detected but must wait, due to the exceptions of its triggered constraint, to the end of its candidacy at which point it becomes detected. This detection results in the cluster denoting the beach to become a subcluster of the cluster associated with the 'dogs' or 'tigers', under feature **location**. In other words, the detection of **onObjectLocation** attaches 'the beach' to the noun phrase which precedes it, under the feature **location**.

This scenario can be complicated if we assume a probable syntactic expectation resulting from a semantic observation: a path could be built between 'talked about' and 'discussed' detecting a similar action. Since the complement of 'talked about' was a topic, it is possible that the reader may expect the complement of 'discussed' to be treated, if possible, as a topic, and therefore, as an NP. In this case, the parsing of 'the (dogs or tigers) on the beach' as an NP is favored through this expectation, and only those interpretations of 'on' that correspond to an NP-attachment become candidates.

The features **onObjectLocation** and **onActionLocation** are simplistic and somewhat inadequate for the more complex second passage. Let us assume instead that the word 'on' triggers more general interpretations of the preposition, such as **onTime** and **onLocation**, reflecting the different possible semantic functions of the preposition. The detection of these features depends on the NP that follows the preposition. Another set of features is used for the attachment of the PP to other parts of the clause. For example, **onLocation** triggers the candidacy of **locationNPAttachment** and **locationVPAttachment**, which respectively attach any location PP to an immediately preceding NP, and any location PP to the verb of the clause. The definitions for these features are similar to those proposed above for **onObjectLocation** and **onActionLocation**. Also, it is possible that even more general features such as **attachPPToNP** and **attachPPToVP**, which would work for any type of PP, could be used (as is the case in the KB of the current prototype). This approach has the advantage of still respecting the 'word expert' philosophy by using a first stratum of features that handle the different semantic functions of a preposition, while modularizing attachment decisions in features not dependent of the particular prepositions but rather associated with general semantic categories (*e.g.*, time, location, etc.).

A possible scenario that uses this more complex disambiguation scheme for 'on' follows:

1. The processing of 'dogs' or 'tigers' leads to a cluster in which those animals are the topic of **actionDiscuss**.

2. The word 'on' leads to the candidacy of the possible interpretations of its semantic function (*e.g.*, **onLocation**, **onTime**, etc.).

3. The words 'the beach' leads to the construction of a cluster that captures the fact that a beach is a location. This cluster is attached to the cluster constructed from the processing of 'on'. Feature **onLocation** becomes detected and triggers the can-

didacy of **locationNPAttachment** and **locationVPAttachment.** The feature **unattachedPP** is placed in the cluster corresponding to the PP in order to indicate the fact that the PP is not currently attached. This feature is deleted from the PP cluster when attachment features such as **locationVPAttachment** become detected.

4. Both of these features may have enough evidence to become detected but must wait until the end of their candidacy, as other possible interpretations are specified as exceptions. This situation captures the structural ambiguity at that point of the processing.

5. The features corresponding to 'while' become detected.

6. The features corresponding to 'sunbathing' become detected. Through the use of 'while', it is inferred that **actionDiscuss** and **actionSunbathe** occur in the same time frame and in the same location. A unique cluster with feature **unspecified** is constructed and made a subcluster of both verbs under feature **location.**

7. The co-occurrence of 'beach' in an unattached location PP and of **actionSunbathe** triggers the inference that the subject of sunbathing performs this action on the beach (which belongs to the time frame of **actionDiscuss**). In order for this inference to become detected, it is required that the beach and the sunbathing be in the same time frame, which is the case. Thus, it is inferred that the sunbathing occurs on the beach. More precisely, the co-occurrence of 'beach' and **action Sunbathe** triggers a buildable feature **sunbatheOnBeach** whose expansion procedure checks that:

   - The word 'beach' belongs to an unattached location PP. This check is performed by testing for the presence of **unattachedPP** and **location** in the cluster associated with the PP.

   - The unattached PP and **actionSunbathe** are in the same time frame (which can be thought of as a cluster whose subclusters under feature **cooccurrentFacts** are the individual actions that occur in this time frame, see chapter 9).

Since these checks are satisfied, **sunbatheOnBeach** becomes detected. It does not modify the context through its expansion procedure, but has **locationVPAttach** as an association. Thus, **locationVPAttach** becomes detected upon receiving the signal from **sunbatheOnBeach**. The unattached PP is attached to **actionSunbathe** under feature **location** and replaces the previous 'unspecified' cluster. Also, other possible interpretations of 'on' are inhibited.

8. Since the **location** subcluster of the **actionDiscuss** cluster is bound to the **location** subcluster of **actionSunbathe**, the attachment to **actionSunbathe** *de facto* realizes the attachment to **actionDiscuss**. In other words, the structural ambiguity is resolved by an inference.

In fact, the inference that the sunbathing occurs on the beach can be erroneous: the women could discuss 'the dogs on the beach' while sunbathing in their garden. If a reader does not make this inference, then the following scenario could continue the previous one:

1. The words 'and then' are processed leading the reader to infer a change of time frame.
2. The words 'went to the cottage' are processed, leading the reader to perceive a change in location for the new time frame. I suggest that it is at this point that the verb attachment can occur. A change of location seems to imply that a previous location was specified. If this isn't the case, and there is an unattached location PP in the previous time frame, then **locationVPAttach** becomes detected.

   In the example, the detection of a change of time and location would expect that the actions of the previous time frame should have a **location** cluster. A check to this effect, triggered by this change in location and time and performed by (the expansion procedure of) feature **needVPLocationInPreviousFrame**, reveals that they do not, leading to the candidacy of feature **findVPLocationInPreviousTimeFrame**. This feature becomes detected if an unattached location PP can be found in the previous time frame. The detection of this feature causes the detection of its association **locationVPAttach**. In other words, in the example, the detection of **findVPLocationInPreviousTimeFrame** leads to the resolution of the structural ambiguity.

The two proposed scenarios not only illustrate **IDIoT**'s treatment of attachment decisions as a lexical disambiguation problem, but also, and most importantly, suggest the omnipresent and complex role of context and inference for linguistic disambiguation. There are still some drawbacks: For example, if no evidence favors either noun or verb attachment, then the user must specify either an order of preference or the detection of an ambiguity. Let us consider an example:

Example 8.3.3 *The women discussed the dogs at breakfast. (Hirst, 1983, p.175)*

Hirst states that his system incorrectly attaches 'at' to 'the dogs' "because the subtle implausibility of the dogs at breakfast as a topic of conversation is not detected" (*ibid.*). In **IDIoT**, since plausibility judgments are not used *per se*, the NP-attachment must be prevented some other way, most likely through an inhibition path that detects this implausibility.

As another example, consider the similar sentence:

Example 8.3.4 *The women discussed the bums at the train station.*

Out of context, this sentence is ambiguous: the women can be in their garden, talking about the bums of the train station, or can be at the train station, discussing bums in general, or can be at the train station discussing the bums of the train station. I leave it to the reader to invent passages in which each of these possibilities would be favored.

Here are some other examples handled by the current prototype of **IDIoT** that mix PP-attachment with reference resolution:

- **Example 8.3.5** *John watched the rabbit in the park.*

  Double attachment of the PP, that is, to the VP and to the NP 'the rabbit'.

- **Example 8.3.6** *John watched the bench in the park.*

  Preference defined between 'bench' and 'park' leads to NP-attachment only.

- **Example 8.3.7** *R1 is a rabbit in a hat. John watched the rabbit in the hat.*

  Reference found for 'the rabbit in the hat' blocks VP-attachment.

- **Example 8.3.8** *John watched the rabbit with the telescope.*

  Double attachment.

- **Example 8.3.9** *John watched the planet with the telescope.*

  Preference defined between 'planet' and 'telescope' leads to NP-attachment only.

- **Example 8.3.10** *R1 is a rabbit. R2 is a rabbit. John watched the rabbit with the hat.*

  No reference resolution, but preference to attach to the NP.

- **Example 8.3.11** *R1 is a rabbit with a telescope. John watched the rabbit with the telescope.*

  Reference found for the NP prevents VP-attachment.

- **Example 8.3.12** *John watched the planet with the microscope.*

  Double attachment.

- **Example 8.3.13** *R1 is a planet with a telescope. R2 is a planet. John watched the planet with the telescope.*

  Reference found for 'the planet with the telescope' prevents VP-attachment.

- **Example 8.3.14** *John eats in a park with a bench.*

  The 'in' PP is attached to 'eats' and, by preference, the 'with' PP is attached to 'park'.

Analytic ambiguities form another facet of the general problem of structural disambiguation and often seem to involve word experts and inferences beyond the scope of syntax. Experimentation with the current prototype of **IDIoT** suggest that, generally, such ambiguities can be viewed as cases of lexical disambiguation or of PP attachment. They will not be discussed furthermore in this dissertation.

Chapter 9

Bridging Inferences

9.1 Inference in Other Models

It is a common view that forming inferences is an essential part of linguistic comprehension. Several theories of inference have been proposed for text comprehension. In this chapter, I investigate the *modus operandi* of inference in **IDIoT** with respect to the most relevant characteristics of these theories.

The first and most important remark is that, as with grammar and disambiguation, I abandon the idea of a correct set of rules of inference that would produce *the correct* inference. Thus, contrary to the majority of the existing models of text comprehension in both psycholinguistics and artificial intelligence, I do not present a theory of what knowledge or representations are involved during inference, or of the rules of composition used to construct inference paths. Instead, I mostly limit this discussion on inference to the role played by the quantitative processes assumed for time-constrained memory. Consequently, the problem of the specificity of features (or equivalently, of the 'granularity' of qualitative dəta) is entirely left to the user. For example, when considering a passage like:

Example 9.1.1 *John is hungry. He picks up the Michelin guide.*

it is each user's responsibility to decide whether the co-occurrence of the features 'person x hungry' and 'person x in possession on Michelin guide' should trigger a direct inference through the feature 'consult Michelin guide in order to satisfy hunger' or requires a more elaborate chain of features (*e.g.*, the goal-based explanation presented in subsection 2.2.2).

Second, Garrod (1985) makes a crucial distinction between a 'true inference', which proceeds from the application of an inferential schema, and a 'pseudo-inference', which arises from interpreting expressions against a mental model. In agreement with other researchers in text understanding (*e.g.*, Graesser and Clark, 1985), he claims that pseudo-inferences are immediate whereas true inferences are only established rarely during comprehension. In other words, most inference paths are constructed from the chaining of existing semantic features or from schema-matching, rather than by the application of formal reasoning rules. Thus, as already stated at the beginning of this part, I do not focus on formal inference but rather on these faster 'pseudo-inferences' typically involved in reading.

Third, recall that existing models in text linguistics[1] (see subsection 2.2.2) can be separated into schema-matching models and inference-chaining models. Moreover, existing computational models, with the exception of Dyer's (1983) work, are restricted to local inferences and ignore the problem of global coherence. Conversely, some models in psycholinguistics do tackle the whole problem of text comprehension (*e.g.*, van Dijk and Kintsch, 1983; Graesser and Clark, 1985), but are still incomplete and may present computational disadvantages.[2] My goal in the rest of this subsection is to suggest that **IDIoT** is a synthesis of these models.

From my standpoint, inference for text comprehension requires both schema-matching *and* inference-chaining as will be argued below. Both mechanisms are implemented at the quantitative level of **IDIoT**. Recall that:

- Schema-matching in **IDIoT** reduces to having the detection of a schema's gate-keeper cause the other schema members to become either detected (through associations) or expected (through expectation signals).
- Inference-chaining in **IDIoT** is implemented by confirmation paths. The proposed mechanism is close in spirit to Hendler's (1986, 1989) work.

An important claim is that more-semantic models of inference can be handled by **IDIoT** if the user wishes to adopt them. Let us consider the three models reviewed at the beginning of chapter 4. Dyer's (1983) approach consists of strict schema matching. Each link between two knowledge structures corresponds to a group of demon processes: whenever a knowledge structure is recognized (*i.e.*, matched), demons are spawned for each of its

[1] Story grammars, discourse analysis, and lexical statistics are not relevant to a discussion on inference *per se* for the simple reason that they do not address this problem.

[2] For example, with respect to the use of probabilistic operations.

links. If a demon fires (*i.e.*, has its constraint satisfied), then the old representation is reinterpreted in terms of the new one. This strategy is similar to the approach suggested for disambiguation using **IDIoT** and is quite easily specifiable. Consider, for example, the 'eat-at-a-restaurant' schema (Dyer, 1983; Norvig, 1987) for which I suggest the following possible partial definitions in **IDIoT**:

```
KU eatAtARestaurant:
% This feature corresponds to the schema of the same name.
 associations: contractualEvent
 constraint 1:
   triggers: actionEat, location, restaurant
   exceptions:
      eatAtFastFood has weight 1
   outputs:
% The following expectations define the schema itself.
      sends expectation signal to waiter
      sends expectation signal to beingSeated
      sends expectation signal to orderingAtRestaurant
      sends expectation signal to payFood
      ...
 expansion:
% I assume that actionEat would have replaced the subject feature
% with the feature eater, as was suggested earlier for the verb
% 'give'. The procedure checks that the eater is a person and that the
% location of actionEat is a restaurant.
      getCluster u1 governing eater
      getCluster u2 governedBy eater in u1
      testPresenceOf person in u2
      getCluster u3 governedBy actionEat in u1
      testPresenceOf location in u3
      getCluster u4 governedBy location in u3
      testPresenceOf restaurant in u4
      renameFeature actionEat to eatAtARestaurant in u1

KU beingSeatedAtRestaurant:
% This feature is a substep of the eat-at-a-restaurant schema.
 constraint 1:
  ordered triggers: eatAtRestaurant, actionSeat
 expansion:
% The procedure checks that the person being seated is the same as the one
% eating. If so, the event of being seated is put under feature 'subSteps'
% in the cluster associated with 'eat-at-a-restaurant': by being governed,
% the substeps become less retrievable and less likely to be stored in LTM.
      getCluster u1 governing seated
      getCluster u2 governedBy seated in u1
      getCluster u3 governing eatAtARestaurant
      getCluster u4 governedBy eater in u3
      testEquivalenceOf u2 u4
      addFeature substeps to u3
      addSubCluster u1 to subSteps in u3

KU waiter:
% This feature is partly defined in terms of the schemata
% that may trigger it.
 constraint c1:
  ordered triggers: eatAtARestaurant, actionServe
 constraint c2:
  ordered triggers: drinkAtBar, actionServe
 ...
 expansion:
% Check that the person being served is also the actor of the triggered
% scenario.
      getCluster u1 governing actionServe
      getCluster u2 governedBy beingServed
      ifConstraint c1 then getCluster u3 governing eatAtARestaurant
      ifConstraint c1 then getCluster u4 governedBy eater in u3
      ifConstraint c2 then getCluster u3 governing drinkAtBar
      ifConstraint c2 then getCluster u4 governedBy drinker in u3
      testEquivalenceOf u2 u4
% If so, add the cluster corresponding to the service performed by the
% waiter to the substeps of the schema.
      addFeature substeps to u3
      addSubCluster u1 to substeps in u3
```

An example using these definitions is discussed in the section 9.3. For now, let me briefly explain the use of feature **substeps** in schema matching. When a schema is detected, it typically sets up expectations. In turn, these are detected if they receive an input signal from all their triggers. The key point is that an expected feature can always be reconstructed from the feature that created the expectation, that is, the one that sent the expectation signal. Furthermore, it is commonly accepted that reconstructable items are less likely to be stored in LTM (*e.g.*, Kintsch and van Dijk, 1978, p.365). Therefore, I suggest that all expected features of a schema that eventually become detected be governed by this schema, under the feature **subSteps**. In other words, the clusters corresponding to the substeps of a schema are subclusters, under **subSteps** of the cluster associated with the schema. As subclusters, they are less reachable that the cluster of the schema itself. Therefore, because the memory manager will select the less reachable elements of STM when reducing the membership of the latter (see chapter 3), the substeps will more likely be forgotten, and thus not be included in the final interpretation.

Norvig's work (1987, p.101) on inference-chaining hinges on the *a priori* definition of a small number of path shapes and inference classes: only a path that matches one of the predefined shapes is considered for evaluation. If so desired, these notions can be readily duplicated in **IDIoT**. Consider, for example, the simplest inference class, namely reference resolution. Recall that an inference class consists of a pair of path shapes that must be matched to the two halves of the total path of a marker collision. For the reference resolution inference, both halves must correspond to a reference path, which is defined as:

$$\text{Reference: } \mathit{origin} \rightarrow I \rightarrow D^{*} \rightarrow \mathit{collision}$$

in which I is an instance relationship, and D, a dominate one (where A dominates B if B is a subclass of A, see *ibid.*, pp.60 and 105). The key observation is that these relationships can be used in **IDIoT** instead of more direct links between KUs. Consider, for example, the following partial definitions:

```
KU johnPerson:
     associations: instanceOfMan, ...
     ...

KU he:
     associations: instanceOfMan, ...

relay KU instanceOfMan:
     suppliers: johnPerson, paulPerson, henryPerson, ...
     customers: man, instanceRelationship

KU man:
     associations: subclassOfPerson, ...
     ...

KU subclassOfPerson:
     suppliers: man, woman, child, ...
     customers: person, subclassRelationship

relay KU instanceRelationship:
     suppliers: instanceOfMan, ...
     customers: refPathShape, ...
```

```
relay KU subclassRelationship:
    suppliers: subclassOfPerson, ...
    customers: refPathShape, ...
```

The feature **johnPerson** will lead, through **instanceOfMan**, to the detection of **instanceRelationship**, which notifies a feature **refPathShape** that would define Norvig's 'reference' path shape. The feature **instanceOfMan** would also eventually lead to the detection of the subclass relationship that would also notify **refPathShape**. Recognizing a sequence of *D* shapes (*i.e.*, a sequence of Dominate links) would be handled in the same say as dealing with a sequence of adjectives (see section 5.1). A very similar approach can be used to capture the arc categories and composition rules suggested by Graesser and Clark (1985, p.74).

There are two points to be made. First, I observe that the distinction between schema-matching and inference-chaining models may be thinner than could be expected. On the one hand, recall that Norvig (1987, p.139) claims that an inference-chaining model can reproduce both script-based and goal-based processing just by looking for connections in the input with respect to what is known in memory. In other words, inference-chaining is taken to subsume schema-matching, though inference-chaining typically does not accommodate expectations. On the other hand, it should be clear that Norvig's approach is very similar to that suggested by Graesser and Clark in that both consist in *matching* path shapes. I suspect that, for non-quantitative models, schema-matching and inference-chaining approaches correspond to distinct, yet equivalent strategies of conceptualization. In the framework of reader-based understanding, I have decided to offer both mechanisms in **IDIoT**: it is up to a specific user to choose the strategy she prefers. Second, it seems that a strictly quantitative approach to inference (*i.e.*, with no built-in semantics), as advocated in Hendler's work (1986, 1989) and in this dissertation, provides a more fundamental level of representation and process over which semantics can be added. It is importanț to realize, as Graesser and Clark (1985, chapter 1) have emphasized, that there is no agreement on the nature and use of inferences. Thus, Dyer's schemas, Norvig's path shapes and inference classes, Graesser and Clark's arc categories and composition rules, and Kintsch and van Dijk's strategies are all experimental rather than definitive. The advantage of a quantitative approach is that these rules are encoded as user-specifiable data rather than being etched in algorithms. Furthermore, the primitive cluster operations supplied for expansion procedures favour a standardization of the specification of these rules. In turn, this promotes the potential re-use and sharing of KUs between users.

A model of the production of inferences alone does not suffice for text comprehension; we require a mechanism to solve the convergence problem, that is, to limit the actual number of inferences generated. In the existing computational models, the problem is either ignored or, in the case of marker-passing systems, handled with magic numbers and anti-promiscuity rules (*e.g.,* Hirst, 1987, section 5.2.3; Hendler, 1986, 1989). A few models emulate memory decay through attrition (*e.g.,* Norvig, 1983b), but there is typically no attempt to model memory *per se.* Conversely, in neurolinguistics (Gigley, 1985a) and psycholinguistics, it is generally accepted that the constraints of the working memory (*i.e.,* capacity and decay) do insure some convergence. Furthermore, as Schnotz (1985) remarks, the construction of inference paths is taken to largely depend on the availability of the relevant features. The mechanisms of time-constrained memory, which underlies **IDIoT**, implement all of these factors:

- Hendler's use of zorch (1989) to limit the length of paths can be directly duplicated with the output specification of a feature (see chapter 3).
- The notions of STM capacity and decay are implemented in the proposed model of time-constrained memory (see chapter 3).
- The issue of the availability of features is handled through the retrievability coefficient of every KU in memory.

An example illustrating the importance of these factors is discussed in section 9.3. Furthermore, the notion of a time-constrained candidacy with a user-specifiable time-span provides an adequate explanation of the reader's natural propensity to infer (see Márkus, 1983), often giving himself merely enough time to consider those elements of STM that are immediately accessible.

On the one hand, the proposed model of time-constrained memory merely forms a first approximation of the complexity of memory processes involved in text comprehension. For example, I do not address some of the more complex features of van Dijk and Kintsch's model of comprehension (1978, pp.368–371; 1983):

- I do not model the kind of processing cycles they assume whereby n_i propositions are processed together. The difficulty is that "the precise number of propositions included in a processing chunk depends on the surface characteristics of the text,... as well as reader characteristics." (*ibid.*).

- As previously mentioned, I do not tackle the tasks of recall and summarization and therefore I do not implement Kintsch and van Dijk's notion of *reproduction probability*. Consequently, I oversimplify the passage of clusters from STM to LTM by not correlating it to frequency of reachability, much as these researchers link the probability of reproduction of x with the number of times x belongs to the STM buffer.

- The use of retrievability coefficients within time-constrained candidacies very roughly approximates the effects of familiarity, which "may be rather complex" (*ibid.*) and involve probabilities of reproduction.

On the other hand, **IDIoT** avoids some of the drawbacks of existing models, especially by enforcing total user-programmability and by not assuming an *a priori* order of processing. For example, there is no need to hypothesize, as Kintsch and van Dijk (1978) do, that inferences and LTM searches are systematically established *after* referential analysis.

As a whole, my model is close in spirit to Graesser and Clark's (1985) work, with the difference that no ordering of operations is imposed contrary to their MBPP algorithm (see section 4.3). In other words, the four operations are independent of one another:

- Schema-matching in **IDIoT** is handled through detections and expectations.

- Bridging is performed in **IDIoT** by inference-chaining (implemented as confirmation paths).

- Projections are expectations set up by detected features, especially schemas.

- Pruning of STM is a concurrent process implemented with the memory manager.

In summary, it appears that **IDIoT** and its underlying model of time-constrained memory can be used to specify the knowledge structures and processes hypothesized in more semantic models of text comprehension.

9.2 Inference with IDIoT

My goal in this section is to suggest how the different facets of inference for text comprehension can be tackled with **IDIoT** on its own, that is, without depending on the specifics of these information-processing models.

Let us start by discussing the functional relationships that exist between clauses, since, I argue, a proper treatment of inference, local coherence, and global coherence ultimately depends on these relationships. The point is that clauses must be related on the basis of their meaning (*i.e.,* intensionally), not just through references (*i.e.,* extensionally). This statement is accepted by Kintsch and van Dijk (1978, pp.390–393) who see this problem as the crucial missing component of their initial work:

> We do not have yet an adequate theory of such functional relations. The present model was not extended beyond the processes involved in referential coherence of texts because we do not feel that the problems involved are sufficiently well understood. However, by limiting the processing model to coherence in terms of argument repetition, we are neglecting the important role that fact relationships play in comprehension.

There are several possible classification schemes for 'facts' (*i.e.,* inter-clausal) relationships:

- From the initial observation that a fact may be a possible, likely, or necessary consequence of another (through connectives such as 'like', 'because', 'although', etc.) Kintsch and van Dijk (1978, p.390) suggest presuppositional relationships: compatibility, enablement, specification, correction, explanation, generalization.
- Some (if not most) of the rules of story grammarians (see subsection 2.2.1) and some of Lehnert's plot units (see subsection 2.2.2) constitute fact relationships.
- Norvig's (1987) inference classes, especially the 'view' class, could be taken as fact relationships.

For a more extensive survey of inference taxonomies refer to Graesser and Clark (1985). The conclusion to be reached is that, as Graesser and Clark state, there is widespread disagreement on inferences in general, and fact relationships in particular.

It is not my intent to develop a theory of these inter-clausal relationships: I repeat that my aim here is not to develop a model of comprehension but to suggest how **IDIoT** can be used to capture such a model. From my standpoint, inter-clausal relationships constitute organizing principles for the construction of clusters. In other words, these relationships essentially specify how the clusters of the facts they relate are organized in STM. I propose distinguishing between local and global organizing principles. Local fact relations typically specify a government link between two clauses. For example, if A is a consequence of B, then there must be a government link between A and B. It is left to the user of **IDIoT** to choose a uniform strategy for each class of fact relation that is to be perceived. For example, if it is established that A is a consequence of B, then A could be systematically made a subcluster of B under the feature **consequence**. Similarly, a correction could always govern the fact that it corrects, and an explained fact, its explanation. It appears that the probability of reproduction is the only guiding principle available to specify these strategies: for example, if it is empirically demonstrated that, for a specific reader, an elaboration x systematically has an smaller likelihood of being recalled than the clause y it elaborates, then x should be made a subcluster of y, for this reduces the reachability of x and thus its chances of passing from STM to LTM. In other words, for this specific reader, each time the **elaboration** feature is detected between two clauses, its expansion procedure makes the elaboration a subcluster of the other fact. In general, given a taxonomy of fact relations and the government strategies adopted by a specific reader, for each of them, **IDIoT** can be used to model the latter, much like the approach suggested to capture Norvig's inference classes.

Finally, it should be remarked that the ordering of the clauses may be very relevant. Consider the sentences:

Example 9.2.1 *Because he is hungry, John eats.*

Example 9.2.2 *John eats because he is hungry.*

It is up to the user of **IDIoT** to decide:

- Whether in both sentences the reason clause ('because he is hungry') governs or is governed by (with respect to reachability) the main clause ('John eats').
- Whether or not the temporal ordering of clauses affects the reproduction probability of each one (*e.g.*, being the first processed, the clause 'because he is hungry' would have better chances of being recalled in the first sentence than in the second).

Whereas local organizing principles specify a government relation between the clusters of two clauses, I propose that global organizing principles would place each cluster on several orthogonal axes of organization. This idea is close to Zavarin's (1983) notion of 'stratification' of levels of representation during comprehension. I suggest that each cluster be, at the very least, placed on the time and location axes. In a complete theory, there could be an axis corresponding to the different semantic roles that can be established within a clause (*e.g.*, actors, time, location, instrument, manner, etc). Kintsch and van Dijk (1978, p.391) make essentially the same hypothesis when they develop the notion of topic change markers (1983; p.204).

Let me very briefly survey the use and implementation of the time and location axes (that were illustrated in the previous chapter). I hypothesize that each of these axes is organized in terms of 'frames'.[3] For example, two facts may be in the same time frame but in the different location frames (*e.g.*, "Yesterday, John did the shopping and Mary played badminton"). Placing a fact on one of these axes means making it a subcluster of the current frame of each of these axes. For both axes, features are required to detect a change of current frame (*e.g.*, different times, different locations). Verb tenses and explicit connectives such as the word 'then' indicate such changes which, if implicit, must be inferred. There is a relatively simple implementation of the notions of axes and current frames in **IDIoT**. An axis is, not surprisingly, a cluster. One feature of an axis identifies the axis itself (*e.g.*, **timeAxis** is a feature of the cluster denoting the time axis). Each of the other features of the cluster governs a set of subclusters belonging to the same frame. The current frame is identified by a specific feature name. For example, to access the current frame on the time axis, one could access the cluster governed by feature **currentTime** in the cluster governing feature **timeAxis**. The user may set up the naming of features so that the previous time frame could also be accessed using the name of its governing feature. Assuming that only the current time frame is directly accessible, the difficulty comes when a change of frame occurs: the set of clusters governed by **currentTime** should now be governed by some feature whose name is irrelevant (since we assume only the current time frame is accessible), and the feature **currentTime** should be made to govern an empty set of clusters. To accommodate the idea of a feature whose name is irrelevant, that is, in essence, a run-time feature that cannot be referred to by the user-specified features of the KB, I introduce the feature **dummyFeature**. By renaming **currentTime** to **dummyFeature**,

[3] This word is used in a totally different sense from its usual one in AI: a frame is merely a set of clusters, an axis of organization consisting of several frames.

the user establishes a boundary between two distinct time frames and makes the current one the only accessible one. Here are sketchy definitions for adding to the current time frame and switching time frames:

```
KU addToCurrentTimeFrame:
 constraint 1:
  triggers: clauseFitsCurrentTime
 expansion:
      getCluster u1 governing clauseFitsCurrentTime
      getCluster u2 governing timeAxis
      getCluster u3 governedBy currentTime in u2
      addSubCluster u1 to currentTime in u3
```

```
KU switchTimeFrame:
 constraint 1:
  triggers: clauseFitsNewTime
 expansion:
       getCluster u1 governing clauseFitsNewTime
       getCluster u2 governing timeAxis
       renameFeature currentTime to dummyFeature in u2
       addFeature currentTime to u2
       addSubCluster u1 to currentTime in u3
```

Both definitions assume that when the ending boundary of a clause is detected, it is possible to infer whether the clause belongs to the current or to a new time frame, to the current or to a new location frame, etc. Some researchers have proposed rules to establish the time relation between two clauses (*e.g.*, Allen, 1982). Also, I repeat that verb tense usages are very important with respect to the time axis. For other possible global axes of organization, there does not seem to be such immediate cues (with the exception of explicit connectives). Given a set of these axes, each clause of a text must be classified with respect to each axis. In other words, the user must specify rules that will decide, for each axis x, whether the current clause being processed belongs to the current frame of x or causes a new frame to be created for x. Since parallelism is postulated, a clause can be placed simultaneously on all the given axes.

Again, the suggested theory for these global organizing principles is simplistic, especially in view of the complexity of the retrieval process(es) that would have to access these axes in order to produce recalls, summaries, and answers to questions on the text read. Yet the importance of these axes should not be underestimated. Consider, for example, the passage:

Example 9.2.3 *John was hungry yesterday afternoon.* [*Otherfacts.*] *Next month, John will pick up a Michelin guide.*

The notation [*Otherfacts.*] denotes a series of zero or more facts. Any path between 'hungry' and 'having a Michelin guide' must be blocked because, it seems, such a path would involve concepts in different time frames. Here is a similar example that involves the location axis:

Example 9.2.4 *John is hungry in Ottawa, while Mary picks up a Michelin guide in Paris.*

If, however, John and Mary were in the same car, then the link between 'hungry' and 'Michelin guide' could be made if John announced he was hungry and Mary picked up the Michelin guide.

There are several points to be made:

- Local organization principles seem to be motivated by a quantitative factor, namely, an ordering between related facts with respect to likelihood of recall.
- Global organization axes establish frame boundaries between different clauses. These boundaries not only must be considered in order to prevent some erroneous inferences, but may also directly correlate to the organization of the representation of a text in memory (and, by extension, of the subsequent retrieval of the representation).
- Thus, both local and global organization principles appear to play an important role in inference, perception of coherence, and, therefore, in comprehension.
- We still lack a theory for these principles and for inference in general.
- A first approximation to a theory of inter-clausal relationships has been presented and can be implemented using **IDIoT**.

Inter-clausal relationships directly correlate with a reader's perception of local and global coherence, and ultimately, with the perception of subject matter. It must be stressed that local and global coherence, and subject matter, are not taken to be perceived one after the other but, on the contrary, simultaneously and interdependently. In particular, local coherence is subject to global constraints, as explained by van Dijk and Kintsch (1983, p.152).

Similarly, both global coherence and subject matter depend on how clauses are interrelated (especially in terms of government) at the local level. Furthermore, in accordance with these researchers, I emphasize the strategic nature of local and global coherence (see van Dijk and Kintsch, 1983, chapters 5 through 7 for details of their strategies).[4]

With regards to the notion of subject matter, Kintsch and van Dijk (*ibid.*, pp.155–6) briefly discuss the notion of sentence topics, which subsumes the more general idea of aboutness:

> [T]he notion of topic can only be appropriately defined in terms of the relations between a sentence and the (con-)text. This is also why such intuitive notions as 'given' and 'new' information, ... have been widely used, although clearly that is not sufficient[.]

The point is that there can be no *a priori* algorithm to compute subject matter, merely idiosyncratic perception involving several factors. I suggest the following non-exhaustive list of items that affect the perception of subject matter (and can be approximately modeled in IDIoT):

- **Topic changes**: In IDIoT, these correspond to global axes for organizing clauses.
- **Inter-clausal government**: Establishing an inter-clausal relationship often implies organizing the concerned clusters into a local government hierarchy. In other words, detecting a relationship between A and B often leads to the cluster of A governing that of B, or *vice versa*.
- **Convergence mechanisms**: In IDIoT, as is generally the case, these mechanisms correspond to the constraints of the short-term memory (*i.e.*, capacity limit and decay). The notion of a time-constrained process is also very important in that it limits, at the quantitative level, the extent of generalizations, expectations, and remindings.[5]
- **Idiosyncratic profile of a reader** (*e.g.*, use of expectations): Several researchers (*e.g.*, Mitchell, 1982, chapter 7; van Dijk and Kintsch, 1983, subsection 2.1.2; Graesser and Clark, 1985) address this item which, from my viewpoint, emphasizes the need for a reader-based approach.

[4] Also, it is commonly accepted that differences between readers are more frequent at the level of global coherence than at the level of local coherence, which involves conventionalized inter-clausal relationships.

[5] This is typically not the case in other models of NLP, in which some arbitrary criterion must be devised to avoid having the representation of a text consist of an endless enumeration of the most irrelevant generalizations (Kintsch and van Dijk, 1978, 1983).

- **Topical inferences:** Though there is no adequate theory of such inferences, it seems a reader can perceive certain word usages and inter-clausal relationships as mechanisms to signal (or highlight) a topic (van Dijk and Kintsch, 1983, chapter 6 and 7).

In conclusion, I repeat Kintsch and van Dijk's observation (1978) that in the general case where readers approach a text with no *a priori* goals or controlling schema, the perception of subject matter can vastly differ from one reader to another.

9.3 Examples of Inter-Clausal Inferences

A schema-matching example was presented in chapter 6.2. The point is that, in essence, a schema defines which actions can be reconstructed (and thus should be governed) by others.

To illustrate the constraints of short-term memory, consider the sentence:

Example 9.3.1 *John is hungry.* [$Otherfacts$.] *John picks up the Michelin guide.*

The notation [$Otherfacts$.] denotes a series of zero or more facts that are not connected to John's hunger (fact F1) nor to John picking up the Michelin (fact F2) in order to find a restaurant. If [$Otherfacts$.] is empty, then the inference will proceed directly with both F1 and F2 in immediate memory. If it contains several items, then F2 will reside in STM but not in WM when the inference is detected. Finally, if the series contains a large number of facts, then F2 will probably have been 'moved' to LTM in which case the inference between F1 and F2 will be missed.

To illustrate local organization, consider the sentence:

Example 9.3.2 *John eats because he is hungry.*

The explicit connective 'because' establishes a government relationship (under feature **rationale**, or **consequence**, or **reason**, etc.) between the two cla'ıses.

And finally, to illustrate the notion of a locational axis of organization, consider the sentence:

Example 9.3.3 *John eats in Ottawa. Mary sleeps in Toronto.*

I suggest the following scenario:

1. Feature **actionEat** possesses a feature **location** that governs the cluster associated with 'Ottawa'.

2. Feature **actionSleep** possesses a feature **location** that governs the cluster associated with 'Toronto'.
3. The feature **clauseFitsDifferentLocation**, which is triggered each time the feature **location** is manipulated (as when a verb has its location specified), tests the difference between the new location and the current one. If these are not the same, the feature detects and notifies **clauseFitsNewLocation** and **clauseFitsPreviousLocation**.
4. The first of these features becomes detected only if the candidacy of the second fails.
5. Feature **clauseFitsPreviousLocation** tries to match the new location with one of the clusters already on the locational axis. If it succeeds, the action (in this case, **actionSleep**) is made a subcluster of the matched location cluster.

9.4 Concluding Remarks

9.4.1 On Semantic Conflicts

It is very probable that, during the processing of a text, erroneous and 'premature' inferences will be drawn, eventually leading the reader to perceive a conflict between a previously established fact and one that has just been detected. I suggest thinking of a conflict in general as a feature (*e.g.*, the disagreement features presented in section 5.1) that becomes detected from the co-occurrence of two (or more) conflicting features. Furthermore, I propose that all features that detect ungrammaticalities (*i.e.*, syntactic conflicts) have **syntacticConflict** as an association, and all features that detect a conflict between two semantic features have **semanticConflict** as an association. The purpose of the features **syntacticConflict** and **semanticConflict** is to localize the actions performed upon the detection of a conflict.

As an example of a semantic conflict, consider the passage:

Example 9.4.1 *John is dead. John eats.*

The co-occurrence of the facts 'person x dead' and 'person x does action' (detected from the action verb 'eat') causes the detection of the feature **deadPersonDoesNotAct** that captures this particular semantic conflict.[6] In turn, **deadPersonDoesNotAct** notifies its association **semanticConflict**.

[6] There is no conflict if John dies after eating. Thus, this conflict can only be detected if it has been somehow acknowledged that the death occurs before the action.

There are several possible strategies to specify a feature like **deadPersonDoesNotAct**. For example, features could be specified so that a person can act only if alive, otherwise a conflict is perceived. I suggest below another approach, which uses a buildable feature to check that the action follows the death once both facts have been detected. The trick is to rely on the fact that if John is dead when he performs an action, then his associated cluster will already have (or govern) feature(s) reflecting this state. In this case, it is the dynamic co-occurrence of conflicting features within the cluster associated with John that will cause the detection of **deadPersonDoesNotAct**. Here is a possible definition:

```
KU deadPersonDoesNotAct:
 associations: semanticConflict
 constraint 1:
  triggers: personActs
% this trigger implies a present action (not a past action).
 expansion:
      getCluster u1 governing clause
      getCluster u2 governedBy subject in u1
% u2 is cluster of the subject.
      testPresenceOf person in u2
% subject must be a person.
      getCluster u3 governedBy personStateAttributes in u2
      testPresenceOf dead in u3
% subject of action is dead
```

This feature is triggered each time a person acts. It checks that the person who acts has the feature **dead** in the set of features corresponding to the **personStateAttributes** cluster associated with each person. A similar approach can be used to check more sophisticated conflicts, such as one where the same person is the subject of two actions taking place at the same time in different locations:

```
KU ubiquitousPerson:
 associations: semanticConflict
 constraint 1:
  triggers: personActs
 expansion:
      getCluster u1 governing clause
      getCluster u2 governing subject in u1
```

```
    testPresenceOf person in u2
    getCluster u10 governing clause
    getCluster u20 governing subject in u10
    testPresenceOf person in u20
% At this point u1 and u10 both govern actions performed by persons.
% Check it's the same person.
    getCluster u3 governedBy person in u2
    getCluster u30 governedBy person in u20
    testEquivalence u3 u30
% Check that the actions occur at the same time.
    getCluster u3 governedBy mainVerb in u1
    getCluster u4 governing time in u3
    getCluster u30 governedBy mainVerb in u10
    getCluster u40 governing time in u30
    testEquivalence u4 u40
% Check for actions occurring in the same location, in which case
% the conflict becomes detected.
% Alternatively, we could use the global time and location axes.
    getCluster u5 governing location in u3
    getCluster u50 governing location in u30
    testEquivalence u5 u50
```

The features **deadPersonDoesNotAct** and **ubiquitousPerson** are extremely specialized. More general rules such as 'an inanimate cannot act' and 'an actor cannot be ubiquitous' could be defined in a similar way; the 'granularity' of the qualitative data specified by the user of **IDIoT** is not the issue here. Indeed, it is commonly accepted that children start with very specialized rules and acquire *some* generalizations with age. Furthermore, a distributed representational scheme makes it quite feasible to have a computationally workable memory with an enormous number of features (distributed over significantly fewer computing elements).

In summary, conflicts are merely a kind of inference that has the peculiarity, in **IDIoT**, of forcing one of the features **syntacticConflict** or **semanticConflict** to become detected. Let us focus on semantic conflicts.

As with other of inferences, there is no single commonly accepted theory of semantic conflicts. What distinguishes such a conflict from another inference is that it somehow needs to be 'resolved'. Kintsch and van Dijk (1978) observe that, upon detecting a semantic conflict, it is typical to either ignore it or to invest more processing resources (*e.g.*, time and memory capacity) in order to resolve it. In the latter case, they add, the increase of resources may increase the conflict's reproduction probability, that is, the likelihood of the conflict being recalled. Granger and Holbrook (1983) suggest a more sophisticated theory: a conflict requires an *inference strategy* in order to be resolved. Given a pre-established and reachable fact x and a conflicting fact y that has just been detected, they propose three possible strategies:

1. **Perseverance**: hold on to x and ignore y.
2. **Recency** : choose y and forget x.
3. **Deferral**: keep both facts and wait for a subsequent resolution.

A fourth strategy could be:

4. **Reconciliation**: invest more resources in order to find a path that could explain the conflict and reconcile the two facts.

More complex phenomena, such as rereading or correcting (as opposed to eliminating) one or both conflicting facts, are not considered here for they involve complex processes of attention and decision that are currently beyond **IDIoT**'s scope of application.

The first three strategies of Granger and Holbrook can be easily modelled in **IDIoT** through the expansion procedure of the **semanticConflict** feature. In the first two cases, the cluster corresponding to the fact to forget is simply deleted. In the third case, the clusters of the conflicting pair of facts could be marked with a special feature (*e.g.,* **unsolvedConflict**), providing access once a resolution has been found within a short amount of time after the conflict has been detected. The fourth case is the trickiest (and probably, the most infrequent) in that it involves requesting more time and memory in order to find an explanatory inference between the two facts. Currently, neither the time-span of a candidacy nor the capacity of STM can be dynamically modified during a reading. In the future, however, a KU should be able to partially alter the time-span of its candidacy, and STM capacity could be increased from any KU through a message to the memory manager.

In conclusion, using **semanticConflict** avoids duplicating a common strategy in individual conflict detectors, but it has the disadvantage of not emulating the mind's ability to dynamically change from one strategy to another while processing a text. If this approach is not satisfactory, the user of **IDIoT** can always specify particular strategies for particular conflict detectors. Overloading an KU with several expansion procedures, one of which would be dynamically chosen (*e.g.*, with respect to the triggered constraint) during processing, is yet another ability to be implemented in **IDIoT** in the future.

9.4.2 Recapitulation of Idiosyncratic Comprehension

As emphasized at the beginning of this part of the thesis, my goal here is not to present a complete and detailed model of reader-based comprehension, but to demonstrate that the quantitative processes of the time-constrained memory provide an adequate framework for developing such a model. For this reason, I have proposed specification strategies for the most frequent problems of linguistic comprehension. Recapitulating, I claim that:

- The notion of time-constrained memory is extremely relevant to the process of real-time parsing (as argued by Lindsay and Manaster-Ramer, 1987).
- The problems of referential resolution, lexical disambiguation, structural disambiguation, and 'figurative' language all critically involve the notion of reachability (of the different possible interpretations) and inference chaining within a short amount of time.
- The processes of inference-chaining and schema-matching, which are essential for both local and global understanding, are provided by **IDIoT**'s underlying model of memory.
- Both local and global organizing principles of a text can be captured with clusters and features.

The point is that a model of comprehension can be specified using **IDIoT**'s trivial algorithm, with the advantage that first, no *a priori* semantic hypotheses are etched in an algorithm, and second, the simple representational scheme of **IDIoT** used for time-constrained memory provides a fundamental quantitative level of representation upon which more complex knowledge structures can be constructed for linguistic comprehension. This research does not address many of the more complex issues of fluent reading (see Mitchell, 1982; van Dijk and Kintsch, 1983). For example, the phenomenon of re-reading is put aside. Such problems often require a theory of self-awareness that essentially eludes current research in psychology and artificial intelligence. Some simpler problems involving attentional processes will eventually be dealt with in **IDIoT**. Let me briefly discuss two of these.

First, Hidi and Baird (1986) report that the perception of interestingness is partly idiosyncratic: "Interest occurs only in the interaction between stimulus and person so that one can never stipulate its origin in one to the exclusion of the other." They also remark that "what is central to the response of interest is that a person is compelled to increase intellectual activity to cope with greater significance of incoming information." I suggest that 'increasing intellectual activity' may correspond to the following actions in **IDIoT**:

- **Increasing STM capacity**: A KU requests from the memory manager that the maximal retrievability coefficient in STM be increased for a short amount of time.
- **Slowing down the rate of decay**: An individual KU could slow down its own rate of decay and/or notify the memory manager that all decay rates should be lowered for a short amount of time.
- **Increasing the time-span of candidacies**: A KU requests the memory manager to increase the time-span of candidacies for a short amount of time.

In essence, we want a KU to have the possibility to dynamically modify the processing environment for a limited period of time. At this point in my research, it seems this may best be accomplished by having the vocabulary for expansion procedures include two new instructions that would respectively increase and decrease intellectual activity as described above. The major drawback of this approach is that special-purpose signals would have to be hypothesized to carry out complex operations. In turn, this points to the need to investigate the actual physiological reification of the currently very metaphorical memory manager.

Second, Kintsch and van Dijk (1978) suggest that the perception of subject matter often relies on the schema encapsulating the reader's goal:[7]

> *Decameron* stories may be read not for the plot and the interesting events but because of concern with the role of women in fourteenth-century Italy or with the attitudes of the characters in the story toward morality and sin.

In other words, the reader has the ability to understand a text from different cognitive viewpoints, different perspectives. I emphasize that these perspectives are totally idiosyn-

[7] This is oversimplified, inasmuch as a reader may simultaneously possess several possibly conflicting schemas. Also, as these authors admit themselves, a reader very often approaches a text with no *a priori* goals, a situation these researchers dismiss as unpredictable, thereby implicitly acknowledging the idiosyncratic nature of linguistic comprehension.

cratic and that each reading of a text may be initiated from a different one. Reading with a perspective corresponds to the frequent situation where a reader comes to a text already seeking a certain aboutness (Vipond and Hunt, 1984). Within the framework of **IDIoT**, I suggest that a perspective consist of a set of features that are initially selected and detected at the beginning of a reading and that reside permanently in WM for this reading. The omnipresence of the perspective in WM guarantees its reachability, and thus its systematic consideration by all candidates.

In accordance with the rest of this part of the dissertation, the point is that quantitative characteristics of **IDIoT** pervade some of the most complex facets of idiosyncratic comprehension. The perception of the subject matter of a text is systematically affected by the metaphors of reachability and of time-constrained memory processes.

Chapter 10

Conclusions

10.1 Recapitulation

Norvig (1987, chapter 7) starts his conclusions by remarking that, "in a sense, FAUSTUS was an experiment in self-deprivation" in that it relied on only six basic inference classes, contrary to systems such as Dyer's (1983) BORIS where "one occasionally gets the suspicion that the system designer can just add one more rule to account for each new difficulty as it arises, as long as he or she is careful about interactions with previous rules" (Norvig, *ibid.*). Yet FAUSTUS does not model the kind of "in-depth understanding" tackled by Dyer. For example, FAUSTUS has no forward inferences, no expectations, no notion of events and explanations. The differences between these two text understanding systems summarize the debate between, on the one hand, researchers looking for general classes of inferences (*e.g.*, for marker-passing models), and on the other hand, researchers constructing a large set of specific rules (for schema-matching models). A multitude of other debates currently exist in text linguistics (*e.g.*, role of priming for words sense disambiguation, declarative *versus* procedural rules, Wilensky's (1983b) principles of comprehension, Graesser and Clark's (1985) principles for a central content selector, etc.).

Given that there is widespread disagreement on the nature, *modus operandi*, and use of inferences in text comprehension, I have tried to avoid specifying, in this dissertation, yet another 'story comprehender' (or conceptual analyzer), that is, yet another set of rules of comprehension. Instead, recognizing that the fundamental feature of conceptual analyzers consists in their ability to build a representation of the input, I suggest a quantitative teuchistic (*i.e.*, 'constructionist') approach to text understanding. This approach is rooted in the basic metaphor of human memory, which bypasses the mind-body problem. I acknowledge the real-time processing constraints imposed by the biological constraint and, therefore, assume that linguistic comprehension is a time-constrained process. I do not model an adaptable memory and, thus, I partition the proposed model of memory into static memory, which consists of a massively parallel network of simple computing elements whose processes allow for the construction of clusters, and dynamic memory, where these clusters reside: Clusters act as the building blocks for the cognitive structures built during comprehension. The implied approach to qualitative data representation and use during comprehension is strictly quantitative in that all data must be specified in terms of constraints to satisfy (through the exchange of simple signals) and sequences of primitive memory operations (expansion procedures). This strategy follows closely the philosophy of Minsky's (1986) "society of mindless agents" from which the mind emerges.

The proposed representational approach is also strictly programmable and reader-based in that all qualitative data is specified by the user of **IDIoT**, who is sole judge of the 'correctness' and 'completeness' of this information. The nature of the rules to use and of the cognitive structures to build during comprehension is entirely left to the user; **IDIoT** merely 'grounds' a theory of text understanding into the more fundamental (representational and processing) level that memory constitutes. From this viewpoint, **IDIoT** is an experiment in *extreme* self-deprivation, in that the designer must develop a conceptual analyzer out of the proposed model of time-constrained memory, which has only six numeric signals and a small set of primitive memory operations. Yet the model of memory has been developed to offer both forward and backward inference chaining, and expectations. And, throughout the preceding chapters, I have argued that **IDIoT** could indeed be used to specify a conceptual analyzer for text understanding.

The representational and processing framework imposed by time-constrained memory does not require the symbolic markers, attenuation mechanisms, and path evaluator of marker-passing systems (see Hendler, 1989), and remedies the severe limitations in generative capacity of existing connectionist systems. Furthermore, through user-specified constraints and buildable features, **IDIoT** makes available both static and dynamic constraints; the latter depending on the contents of STM at a particular point in time. The notion of a time-constrained process also eliminates the issues of algorithmic complexity and intractability, as time is the stopping criterion for comprehension. Similarly, given that some inferences take longer to establish than others, the time limit of memory processes implicitly controls the contents of the final representation of the input (to be used by an eventual recall mechanism to summarize and answer questions about what was read).

It could seem that having arbitrarily long delays for memory processes would necessarily lead to more 'in-depth' understanding. However, since short-term memory constantly decays, this is not the case, for the longer the delay, the more clusters in STM will decay to the point of being moved to LTM or forgotten. There is another trade-off associated with long delays: such delays do allow less retrievable information to be retrieved, but the longer the delay, the more 'far-fetched' retrieved information can get. In other words, the presence in STM of less 'relevant' information increases with time. But, ultimately, it is the designer, through the specification of rules, who may define what is relevant and what is not. From this standpoint, **IDIoT** offers a *tabula rasa* approach to cognition: there are only quantitative processes and their limitations, much like the physiological limitations of the brain, and 'mind' (in essence, the **IDIoT** knowledge base) starts empty. It is entirely left to the user of **IDIoT** to define rules and organize their interactions. Indeed, inconsistent sets of rules can exist in the KB: the *actual* set of rules used at a given point in time mainly depends on the reachability of rules at that point in time (Lindsay and Manaster-Ramer, 1987). In other words, in **IDIoT**, as the number of rules in the KB increases, the issue is not as much defining interactions between rules as specifying communication delays that give precedence to certain rules under certain circumstances. Indeed, much as connectionist researchers 'tweak' weights on connections, the user of **IDIoT** must 'tweak' the communication delay between two knowledge units to ensure that one rule is triggered before or after another one, or to leave enough time for a KU to force the detection or inhibition of another one. Timing is of the essence in **IDIoT**.

10.2 Enhancements

Several enhancements will be implemented for the next prototype of **IDIoT**:

- A KU will have the ability to increase or decrease the retrievability of its customers. This will make the whole detection mechanism far more dynamic in that the actual reading process will affect the likelihood of the detection of a feature through the detection of other features.

- The memory manager will warn the user if two KUs that are mutual exceptions are simultaneously present in STM. This warning will make the user aware that either there is an error in the KB or there is a potential conflict to explicitly handle.

- Static and dynamic memory will be more integrated. Roughly put, a KU will act as a 'computing cluster', that is, behave as a cluster once it is activated. More specifically, if the user wishes so, upon its activation, a KU will be 'moved' to the Working Memory and will not be 'deactivated' until it is removed from STM. And a KU in STM will be considered far more retrievable than KUs in the knowledge base. Thus, in effect, a KU will have its retrievability (and 'detectability') controlled by its actual usage. Again, this strategy makes **IDIoT** significantly more dynamic.

- The knowledge browser will have a new menu item to obtain a graphical layout of the knowledge base.

- A tool to navigate in the KB will be designed. From a KU, the designer will be able to easily access the suppliers and customers of this KU.

- The syntactic and semantic checking of expansion procedures will be improved.

- Each KU in static memory will be assigned an initial activation level. The higher this level, the slower the decay rate of the KU once activated. This follows from the observation (*e.g.*, Gernsbacher, 1985) that thematic information takes longer to become detected than syntactic information but that former is more permanent in memory than the latter.

- To implement the notion of a perspective (see chapter 9), the user will be able to mark a set of KUs as being permanently detected for a reading and to construct clusters that reside in STM throughout a reading.

- Through two new primitive memory operations, an expansion procedure will be able to increase or decrease, for a short amount of time, the detection threshold, the length of races, and the capacity of memory. This will allow the user to simulate a fluctuation in interestingness (see chapter 9).

- A KU will be allowed to have several expansion procedures, each associated with one or more constraints. At the time of its detection, the KU will execute the expansion procedure associated with its triggered constraint. This ability will make it easy for the user to regroup several features in a same KU.

- There will be different levels of tracing in order to avoid having one large and very verbose log of all exchanges of signals and modifications of states during a reading.

I also want to investigate the 'true' parallel execution of expansion procedures. **IDIoT** should ideally run on a parallel computer that supports asynchronous communication (which is *not* the case for the Connection Machine).

10.3 Future Research

The model of time-constrained memory presented in this work has some important gaps that future research must address.

With respect to learning:

- Since I do not model an adaptable memory, there is no principle of self-organization in **IDIoT**. Such a principle (*e.g.*, from category theory, see Peters and Shapiro (1987) and Peters, Shapiro, and Rapaport (1989)) would be desirable if it proceeded from quantitative observations (*e.g.*, with respect to the physiology of the brain/mind).

- Learning from experience should be integrated within **IDIoT**. More specifically, without necessarily developing a complete theory of learning, it should be possible for **IDIoT** to continuously monitor (through the memory manager) the contents of STM for new patterns of features to be suggested to the reader for addition to the KB. In its simplest form, this would involve keeping a frequency count of the co-occurrence, in STM, of 'unrelated' features. In the longer term, an automatic generalization mechanism would also be highly desirable.

- The distinction between static and dynamic memory should be eliminated in favor of the kind of fast and slow links suggested by Hinton and Plaut (1987).

- Ideally, the system should be able to automatically generate constraints for the detection of new unknown features, though this seems quite far-off for now.

With respect to memory management:

- A STM capacity limit defined in terms of maximal membership is simplistic. Several management schemes have been proposed in both psychology and artificial intelligence. Lehnert's (1981, 1983) notion of pivotal units seems the easiest to implement in **IDIoT**.

- The mechanisms of information loss, especially the "shift-processing hypothesis" (Gernsbacher, 1985) need further investigation. The phenomenon of forgetting needs to be accounted for in **IDIoT**, most likely through the idea that only clusters with more than a certain threshold of energy move from STM to LTM; others are simply lost.

- The current retrieval mechanism of **IDIoT**, though relatively complex, is oversimplified compared to theories proposed in psychology (*e.g.*, the notion of synergy for engrams, see Tulving, 1983, 1984). These theories must be studied at greater length so that, eventually, it would be possible for the user of **IDIoT** to define a retrieval mechanism. More generally, I would like the memory model of **IDIoT** to be itself somewhat user-programmable.

- The notions of strategic reading (van Dijk and Kintsch, 1983) and strategic inferences (Granger and Holbrook, 1983) must be integrated in **IDIoT**. In particular, the user should be able to specify a strategy for the processing of conflicting inferences.

- The notion of the acceptability of an interpretation must be investigated. This topic partly relates to the treatment of surprisingness.

Also, the problems of recall, summarization, and question answering must be addressed. In particular, the notion of a central content selector (Graesser and Clark, 1985) must be investigated for **IDIoT**. For question answering, a simple strategy could consist in constructing, from the processing of the question, a cluster to be matched in the representation built for the studied text.

10.4 Aftermath

IDIoT started out and remains nothing more than a tool to explore how much of comprehension depends on the time-constrained nature of human memory. By all means, the proposed model of memory is simplistic but already helpful in demonstrating the omnipresent and determinative role of time during comprehension. Since all the complexities of the implementation of time-constrained memory are invisible to the user of IDIoT, it is easy to believe that the trivial algorithm, that is, the *modus operandi* of time-constrained memory, is just that, trivial, and that the interpretation of a text results solely from the rules of the KB, IDIoT merely acting as an access system. This is the danger of any AI research where the focus of study is shifted away from conceptual rules. Yet, in light of the numerous debates raging with respect to linguistic comprehension, I do believe we must 'ground' the data structures and algorithms we postulate for cognition in a more fundamental entity that bypasses the mind-body problem, namely, human memory. This is precisely what I tried to do in the previous chapters, to show that both new and existing strategies for understanding can be specified using IDIoT. But, regardless of the exact rules used, I also want to argue that the time-constrained nature of memory plays an essential role during comprehension by defining context and controlling the interactions between rules (*e.g.*, through the processes of decay, memory management, races, retrieval, etc.). The problems associated with the specification of rules (see chapter 2) do not vanish in IDIoT, but some may be partially avoided. For example, a parallel network may help with respect to intractability, and the notion of reachability does considerably limit the interactions that a designer must consider when adding a KU to the knowledge base.

In the end, the specification of rules and concepts for the knowledge base remains a crucial and enormous task for which different approaches can be adopted. Most likely, human knowledge is made of a mixture of both 'experts' and general rules abstracted from experience. I do not deny this evidence; I merely claim that these rules and their interactions should be grounded in memory. It is obvious that a very large amount of work still must be done before we can construct a reasonable model of human memory (*e.g.*, with respect to natural and analogical categorization, 'body knowledge', and information loss) and design unconstrained memory-based conceptual analyzers for text comprehension. This is the direction of my future efforts.

Bibliography

[1] Alba, J.W. and Hasher, L. (1983) "Is Memory Schematic?", *Psychological Bulletin*, 93:203–231.

[2] Adler, Mortimer (1985) *Ten Philosophical Mistakes*, Macmillan Publishing Company, New York.

[3] Ajjanagadde, Venkat and Shastri, Lokendra (1989) "Efficient Inference with Multi-Place Predicates and Variables in a Connectionist System", *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, Ann Arbor, MI.

[4] Allen, James (1982) "Modelling Events, Actions, and Time", *Proceedings of the Fourth Annual Conference of the Cognitive Science Society*, Ann Arbor, pp.5–6.

[5] Alterman, Richard (1985) "A Dictionary Based on Concept Coherence". *Artificial Intelligence*, 25: 153–186.

[6] Alshawi, Hiyan (1987) *Memory and Context for Language Interpretation*, Cambridge University Press, Cambridge, England.

[7] Anandan, P., Letovsky, S. and Mjolsness, E. (1989) "Connectionist Variable-Binding by Optimization", *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, Ann Arbor, MI.

[8] Anderson, John R. (1980) *Cognitive Psychology and Its Implications*, W. H. Freeman and Company, San Francisco, CA.

[9] Anderson, John R. (1983) *Architecture of Cognition*, Harvard University Press, Cambridge, MA.

[10] Arbib, Michael, Conklin, Jeffrey and Hill, Jane (1987) *From Schema Theory to Language*, Oxford University Press, New York.

[11] Baddeley, Alan (1976) *The Psychology of Memory*, Basic Books, New York.

[12] Baddeley, Alan (1986) *Working Memory*, Oxford University Press, New York.

[13] Barnden, John (1983) "On Association Techniques in Neural Representations Schemes", *Proceedings of the Fifth Annual Conference of the Cognitive Science Society*, Rochester, New York.

[14] Barthes, Roland (1977) *Roland Barthes by Roland Barthes*, Translation by Richard Howard, Macmillan, London, England.

[15] Beaugrande, Robert de (1980) *Text, Discourse, and Process*, Ablex, Norwood, NJ.

[16] Berg, George (1987) "A Parallel Natural Language Processing Architecture with Distributed Control", *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, Seattle, WA, pp.487–495.

[17] Berlin, Brent and Kay, Paul (1969) *Basic Color Terms: Their Universality and Evolution*, University of California Press, Berkeley, CA.

[18] Birnbaum, Lawrence (1985) "Lexical Ambiguity as a Touchstone for Theories of Language Analysis", *Proceedings of the Ninth Joint Conference on Artificial Intelligence*, Los Angeles, California, pp.815–820.

[19] Birnbaum, Lawrence (1989) "A Critical Look at the Foundations of Autonomous Syntactic Analysis", *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, Ann Arbor, MI, pp.99–106.

[20] Braine, Martin (1978) "On the Relation Between the Natural Logic of Reasoning and Standard Logic", *Psychological Review*, 85: 1–21.

[21] Bransford, John and Johnson, Marcia (1973) "Considerations of Some Problems of Comprehension", In: Chase, William (ed.), *Visual Information Processing*, Academic Press, New York, pp.383–435.

[22] Britton, Bruce and Black, John (1985) *Understanding Expository Text*, Lawrence Erlbaum Associates, Hillsdale, NJ.

[23] Charniak, Eugene (1983) "Passing Markers: A Theory of Contextual Influence in Language Comprehension", *Cognitive Science*, 7:171–190.

[24] Charniak, Eugene (1986a) "A Single-Semantic-Process Theory of Parsing", Technical Report, Department of Computer Science, Brown University, Providence, RI.

[25] Charniak, Eugene (1986b) "A Neat Theory of Marker Passing", *Proceedings of the National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, Philadelphia, Pennsylvania, pp.584–588.

[26] Charniak, Eugene and Goldman, Robert (1988) "A Logic for Semantic Interpretation", *Proceedings of the 26th Meeting of the ACL*, Buffalo, pp. 87–94.

[27] Chomsky, Noam (1965) *Aspects of the Theory of Syntax*, MIT Press, Cambridge, MA.

[28] Chomsky, Noam (1980) *Rules and Representations*, Columbia University Press, New York.

[29] Chomsky, Noam (1982) *Some Concepts and Consequences of the Theory of Government and Binding*, MIT Press, Cambridge, MA.

[30] Chun, Hon Wai and Mimo, Alejandro (1987) "A Model of Schema Selection Using Marker Passing and Connectionist Spreading Activation", *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, Seattle, WA, pp.887–896.

[31] Churchland, Paul (1985) "Eliminative Materialism and the Propositional Attitudes", *Journal of Philosophy*, 73:67–90.

[32] Conrad, Michael (1985) "On Design Principles for a Molecular Computer", *Communications of the ACM*, 28:464–480.

[33] Corriveau, Jean-Pierre (1987) "On the Role of Time in Reader-Based Comprehension", *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, Seattle, WA, pp.794-801.

[34] Corriveau, Jean-Pierre (1988) review of Alshawi (1987), In: *Computational Linguistics*, July.

[35] Cottrell, Garrison (1984) "A Model of Lexical Access of Ambiguous Words", *Proceedings of the National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, Austin, Texas, pp.61–67.

[36] Cottrell, Garrison (1989) *A Connectionist Approach to Word Sense Disambiguation*, Morgan Kaufmann, San Mateo, CA.

[37] Culler, Jonathan (1975) *Structuralist Poetics*, Routledge and Kegan Paul, London, England.

[38] Cullingford, R. (1978) *Script Applications: Computer Understanding of Newspaper Stories*, Ph.D. thesis, Yale University, Department of Computer Science, New Haven, CT.

[39] Dascal, Marcelo "On the Roles of Context and Literal Meaning in Understanding", *Cognitive Science*, 13:253–258.

[40] DeJong, G. (1982) "An Overview of the FRUMP System", In: Lehnert, W. and Ringle, M. H. (eds.), *Strategies for Natural Language Processing*, Lawrence Erlbaum Associates, Hillsdale, NJ.

[41] Derthick, Mark and Plaut, David (1986) "Is Distributed Connectionism Compatible with the Physical Symbol System Hypothesis?", *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, Amherst, Massachusetts, pp.639–644.

[42] Dillon, George (1980) "Discourse Processing and the Nature of Literary Narrative", *Poetics* 9:163–180.

[43] van Dijk, Teun A. (1980) "Story Comprehension: An Introduction", *Poetics*, 9:1–21.

[44] van Dijk, Teun A. and Kintsch Walter (1983) *Strategies of Discourse Comprehension*, Academic Press, New York.

[45] Dyer, Michael G. (1983) *In-Depth Understanding*, MIT Press, Cambridge, MA.

[46] Elman, Jeffrey (1989) "Structured Representations and Connectionist Models", *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, Ann Arbor, MI.

[47] Eiselt, Kurt and Granger, Richard (1987) "A Time-Dependent Distributed Processing Model of Strategy-Driven Inference Behavior", *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, Seattle, WA.

[48] Etherington, David and Reiter, Raymond (1985) "On Inheritance Hierarchies With Exceptions", In: Brachman, Ronald and Levesque, Hector (eds.), *Readings in Knowledge Representation*, Morgan Kaufmann,Los Altos, CA.

[49] Fahlman, S.E. (1979) *NETL: A System for Representing and Using Real-World Knowledge*, MIT Press, Cambridge, MA.

[50] Feldman, Jerome (1984) "Computational Constraints from Biology", *Proceedings of the Sixth Annual Conference of the Cognitive Science Society*, Boulder, Colorado, p.101.

[51] Feldman, Jerome (1985a) "Connectionist Models and Their Applications: Introduction", *Cognitive Science*, 9:1–2.

[52] Feldman, Jerome (1985b) "Energy and the Behavior of Connectionist Models", Technical Report no.155, Department of Computer Science, University of Rochester, Rochester, NY.

[53] Firth, J.R. (1957) "A Synopsis of Linguistic Theory, 1930–1950", In: Firth, J.R. (ed.), *Studies in Linguistic Analysis*, Basil Blackwell, Oxford, England.

[54] Fodor, Janet and Frazier, Lyn (1980) "Is the Human Sentence Parsing Mechanism an ATN?", *Cognition*, 8:417–459.

[55] Fodor, Jerry and Pylyshyn, Zenon (1988) "Connectionism and cognitive architecture: A critical analysis", *Cognition*, 28:3-71.

[56] Frazier, Lyn and Fodor, Janet (1978) "The Sausage Machine: A new two-stage parsing model", *Cognition*, 6:291-325.

[57] Gadamer, Hans-Georg (1976) *Philosophical Hermeneutics*, translated by David Linge, University of California Press, Berkeley, CA.

[58] Gardner, Howard (1983) *Frames of Mind: The Theory of Multiple Intelligences*, Basic Books, New York.

[59] Garnham, Allan (1983) "What's Wrong with Story Grammars", *Cognition*, 15:145–154.

[60] Garrod, Simon C. (1985) "Incremental Pragmatic Interpretation versus Occasional Inferencing during Fluent Reading", In: *Inferences in Text Processing*, Rickheit, G. and Strohner, H. (eds.), Elsevier Science Publishers (North Holland), Amsterdam.

[61] Gernsbacher, Morton Ann (1985) "Surface Information Loss in Comprehension", *Cognitive Psychology*, 17:324–363.

[62] Gibbs, Raymond (1984) "Literal Meaning and Psychological Theories", *Cognitive Science*, 8:275–304.

[63] Gibbs, Raymond (1989) "Understanding and Literal Meaning", *Cognitive Science*, 13:243–252.

[64] Gigley, Helen (1985a) "Computational Neurolinguistics: What Is It All About?", *Proceedings of the Ninth Joint Conference on Artificial Intelligence*, Los Angeles, California, pp.260–266.

[65] Gigley, Helen (1985b) 'Grammar Viewed as A Functional Part of a Cognitive System", *Proceedings of the 22nd Conference of the Association for Computational Linguistics*, Chicago, Illinois, pp.324–332.

[66] Goldberg, Adele (1984) *SMALLTALK-80: The Interactive Programming Environment*, Addison-Wesley, Reading, MA.

[67] Gorfein, David (1989) *Resolving Semantic Ambiguity*, Springer-Verlag, New York.

[68] Graesser, Arthur and Clark, Leslie (1985) *Structures and Procedures of Implicit Knowledge*, Ablex Publishing Corporation, Norwood, NJ.

[69] Granger, Richard and Holbrook, Jennifer (1983) "Perseverers, Recencies and Deferrers: New Experimental Evidence for Multiple Inference Strategies in Understanding", *Proceedings of the Fifth Annual Conference of the Cognitive Science Society*, Rochester, NY.

[70] Granger, Richard, Holbrook, Jennifer and Eiselt, Kurt (1983) "STRATEGIST: A Program that Models Strategy-Driven and Content-Driven Inference Behavior", *Proceedings of the National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, Washington, DC.

[71] Grosz, Barbara J. (1977) *The Representation and Use of Focus in Dialogue Understanding*, Ph.D. Thesis, SRI Technical Note no.151, SRI International, Menlo Park, CA.

[72] Habel, Christopher (1983) *Stories: An Artificial Intelligence Perspective*, Research Report no.19, Technische Universität Berlin, Berlin.

[73] Haberlandt, Karl (1980) "Encoding of Story Constituents", *Poetics*, 9:99–116.

[74] Haddock, Nicolas (1987) "Incremental Interpretation and Combinatory Categorial Grammar", *Proceedings of the Tenth Joint Conference on Artificial Intelligence*, Milan, Italy, pp.661-663.

[75] Hendler, James (1987) *Integrating Marker-Passing and Problem-Solving: A Spreading-Activation Approach to Improved Choice in Planning*, Ablex, Norwood, NJ.

[76] Hendler. James (1989) "Marker-Passing over Microfeatures: Towards a Hybrid Symbolic/Connectionist Model", *Cognitive Science*, 13:79–106.

[77] Hidi, Suzanne and Baird, William (1986) "Interestingness—A Neglected Variable in Discourse Processing", *Cognitive Science*, 10:179–194.

[78] Hinton, Geoffrey and Plaut, David (1987) "Using Fast Weights to Deblur Old Memories", *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, Seattle, WA.

[79] Hinton, Geoffrey and Sejnowski, Terry (1984) "Learning Semantic Features", *Proceedings of the Sixth Annual Conference of the Cognitive Science Society*, Boulder, Colorado, pp.63–70.

[80] Hirsch, E.D. Jr. (1967) *Validity in Interpretation*, Yale University Press, New Haven, CT.

[81] Hirst, Graeme (1981) *Anaphora in Natural Language Understanding*, Springer-Verlag, New York.

[82] Hirst, Graeme (1984) "Jumping to Conclusions: Psychological Reality and Unreality in a Word Disambiguation Program", *Proceedings of the Sixth Annual Conference of the Cognitive Science Society*, Boulder, Colorado, pp.179–182.

[83] Hirst, Graeme (1987) *Semantic Interpretation and the Resolution of Ambiguity*, Cambridge University Press, Cambridge, England.

[84] Hirst, Graeme (1988a) "Resolving lexical ambiguity computationally with spreading activation and Polaroid Words." In: Small, Steven, Cottrell, Garrison, and Tanenhaus, Michael (eds.) *Lexical ambiguity resolution*, Morgan Kaufmann, 1988, Los Altos, CA, pp.73–106.

[85] Hirst, Graeme (1988b) "Semantic Interpretation and Ambiguity", *Artificial Intelligence*, 34:131–177.

[86] Hobbs, Jerry R., Stickel, Mark, Martin, Paul and Edwards, Douglas (1988) "Interpretation as Abduction", *Proceedings of the 26th Annual Meeting of the ACL*, Buffalo.

[87] Holub, Robert (1984) *Reception Theory—A Critical Introduction*, Methuen, New York.

[88] Ide, Nancy (1986) "A Computational Approach to Meaning in Literary Texts", *Proceedings of the Conference on Computing and the Humanities: Today's Research, Tomorrow's Teaching*, Toronto, pp.222–228.

[89] Johnson, Mark (1987) *The Body in the Mind: The Bodily Basis of Meaning, Imagination, and Reason*, University of Chicago Press, Chicago, IL.

[90] Johnson, Nancy and Mandler, Jean (1980) "A Tale of Two Structures: Underlying and Surface Forms in Stories", *Poetics*, 9:51–86.

[91] Johnson-Laird, Philip (1982) "Thinking as a Skill", *Quarterly Journal of Experimental Psychology*, 34A: 1–29.

[92] Kass, Alex (1986) "Modifying Explanations to Understand Stories", *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, Amherst, Massachusetts, pp.691–696.

[93] Kintsch, Walter (1980) "Learning from Text, Levels of Representation, or: Why Anyone Would Read a Story Anyway", *Poetics*, 9:87–98.

[94] Kintsch, Walter and van Dijk, Teun (1978) "Towards a Model of Text Comprehension and Production", *Psychological Review*, 85:363–394.

[95] Kintsch, Walter and van Dijk, Teun (1983) Towards a Model of Strategic Discourse Processing, Academic Press Inc., New York, NY.

[96] Koestler, Arthur (1978) *Janus: A Summing Up*, Vintage Books, New York.

[97] Lange, Trent and Dyer, Michael (1989) "'Frame Selection in a Connectionist Model of High-Level Inferencing", *Proceedings of the Eleventh Conference of the Cognitive Science Society*, Ann Arbor, MI.

[98] Lange, Trent, Hodges, Jack, Fuenmayor, Maria, Belyaev, Leonid (1989) "DESCARTES: Development Environment for Simulating Hybrid Connectionist Architectures", *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, Ann Arbor, MI, pp.698–705.

[99] Lakoff, George (1987) *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*, University of Chicago Press, Chicago, IL.

[100] Lakoff, George and Johnson, Mark (1980) *Metaphors We Live By*, University of Chicago Press, Chicago, IL.

[101] Leake, David (1989) "Anomaly Detection Strategies for Schema-Based Story Understanding", *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, Ann Arbor, MI.

[102] Leake, David and Owens, Christopher (1986) "Organizing Memory for Explanation", *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, Amherst, Massachusetts, pp.710–715.

[103] Lebowitz, Michael (1980) "Generalization and Memory in an Integrated Understanding System", Technical Report no.186, Department of Computer Science, Yale University, New Haven, CT.

[104] Lebowitz, Michael (1988) "The Use of Memory in Text Processing", *Communications of the ACM*, 31:1483–1501.

[105] Lee, Geunbae, Flowers, Margot, Dyer, Michael (1989) "A Symbolic/Connectionist Script Applier Mechanism", *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, Ann Arbor, MI.

[106] Lehnert, Wendy (1981) "Plot Units and Narrative Summarization", *Cognitive Science*, 5:293–332

[107] Lehnert, Wendy (1983) "Narrative Complexity Based on Summarization Algorithms", *Proceedings of the Eighth Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany, pp.713–716.

[108] Lindsay, Robert and Manaster-Ramer, Alexis (1987) "Teuchistic Natural Langugage Processes", *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, Seattle, WA., pp.96–105.

[109] Lytinen, Steven (1984) *The Organization of Knowledge in a Multi-Lingual, Integrated Parser*, Ph.D. Thesis, Yale University, Department of Computer Science, New Haven, CT.

[110] Mac Cormac, Earl (1985) *A Cognitive Theory of Metaphor*, MIT Press, Cambridge, MA.

[111] Malsburg, Charles von der (1985) "Algorithms, Brain, and Organization", In: Demongeot, J., Golès, E. and Tchuente, M. (eds.) *Dynamical Systems and Cellular Automata*, Academic Press, London.

[112] Marcus, Mitchell P. (1980) *A Theory of Syntactic Recognition for Natural Language*, MIT Press, Cambridge, MA.

[113] Marcus, Mitchell P. (1984) "Some Inadequate Theories of Human Language Processing", In: Bever, Thomas, Carroll, John and Miller, Lance (eds.), *Talking Minds*, MIT Press, Cambridge, MA.

[114] Márkus, András (1983) "Shifting the Focus of Attention", *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany, pp.66–68.

[115] Martin, Charles (1989) "Pragmatic Interpretation and Ambiguity", *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, Ann Arbor, MI.

[116] McClelland, James L. and Kawamoto, Andrew H. (1986) "Mechanisms of Sentence Processing: Assigning Roles to Constituents", In: McClelland, James and Rumelhart, David (eds.) *Parallel Distributed Processing*, vol. 2, MIT Press, Cambridge, MA.

[117] McClelland, James and Rumelhart, David (1986) *Parallel Distributed Processing*, vol. 2, MIT Press, Cambridge, MA.

[118] McRoy, Susan (1988) *The Influence of Time and Memory Constraints on the Resolution of Structural Ambiguity*, Technical Report CSRI-209, University of Toronto, Department of Computer Science, Toronto, Canada.

[119] van der Meer, Elke (1987) "Mental Representation of Events", In: van der Meer, E. and Hoffman, J. (eds.), *Knowledge-Aided Information Processing*, North-Holland, Amsterdam.

[120] Miezitis, Mara (1988) *Generating Lexical Options by Matching in a Knowledge Base*, Technical Report CSRI-217, University of Toronto, Department of Computer Science, Toronto, Canada.

[121] Miller, George (1985) "Dictionaries of the Mind", *Proceedings of the 22nd Conference of the Association for Computational Linguistics*, Chicago, Illinois, pp.305–314.

[122] Minsky, Marvin (1975) "A Framework for Representing Knowledge", In: Winston, Patrick (ed.), *The Psychology of Computer Vision*, McGraw-Hill, New York.

[123] Minsky, Marvin (1986) *The Society of Mind*, Simon and Schuster, New York.

[124] Mitchell, D.C. (1982) *The Process of Reading: A Cognitive Analysis of Fluent Reading and Learning to Read*, John Wiley & Sons, New York.

[125] Muller, Charles (1977) *Principes et méthodes de statistique lexicale*, Hachette, Paris.

[126] Norris, Christopher (1982) *Deconstruction—Theory and Practice*, Methuen, New York.

[127] Norris, Dennis (1986) "Word Recognition: Context Effects without Priming", *Cognition*, 22: 93-136.

[128] Norvig, Peter (1983a) "Six Problems for Story Understanders", *Proceedings of the National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, Washington, D.C., pp.284–287.

[129] Norvig, Peter (1983b) "Frame Activated Inferences in a Story Understanding Program", *Proceedings of the Eighth Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany, pp.624–626.

[130] Norvig, Peter (1987) *Unified Theory of Inferences for Text Understanding*, Ph.D. Thesis, University of California at Berkeley, Department of Computer Science, Berkeley, CA.

[131] Norvig, Peter (1989) "Marker Passing as a Weak Method for Text Inferencing", *Cognitive Science*, 13:569–620.

[132] Odell, Jack (1984) "On the Possibility of Natural Language Processing: Some Philosophical Objections", *Theoretical Linguistics*, 11:127–146.

[133] Peckham, Morse (1979) *Explanation and Power—The Control of Human Behaviour*, Seabury, New York.

[134] Peters, Sandra and Shapiro, Stuart (1987) "A Representation for Natural Category Systems", *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, Seattle, WA, pp.379–390.

[135] Peters, Sandra, Shapiro, Stuart and Rapaport, William (1988) "Flexible Natural Language Processing and Roschian Category Theory", *Proceedings of the Tenth Annual Conference of the Cognitive Science Society*, Montréal, Canada, pp.125–131.

[136] Phillips, Martin (1985) *Aspects of Text Structure*, North Holland, Amsterdam.

[137] Piaget, Jean (1970) *Genetic Epistemology*, Columbia University Press, New York.

[138] Piatelli-Palmarini, Massimo, ed. (1980) *Language and Learning*, Harvard University Press, Cambridge, MA.

[139] Pinker, Steve and Prince, Alan (1988) "On Language and Connectionism: Analysis of a Parallel Distributed Processing Model of Language Acquisition", *Cognition*, 28:73–193.

[140] Plantinga, Edwin (1986) "Who Decides What Metaphors Mean?", *Proceedings of the Conference on Computing and the Humanities: Today's Research, Tomorrow's Teaching*, Toronto, pp.194–204.

[141] Plantinga, Edwin (1987) "Mental Models and Metaphor", *Proceedings of Theoretical Issues in Natural Language Processing*, Las Cruces, New Mexico, pp.164–172.

[142] Pollack, Martha E. and Pereira, Fernando C.N. (1988) "An Integrated Framework for Semantic and Pragmatic Interpretation", *Proceedings of the 26th Annual Meeting of the ACL*, State University of New York, Buffalo.

[143] Propp, Vladimir (1968) *Morphology of the Folktale*, University of Texas Press, Austin.

[144] Pylyshyn, Zenon (1984a) *Computation and Cognition: Toward a Foundation of Cognitive Science*, MIT Press, Cambridge, MA.

[145] Pylyshyn, Zenon (1984b) "Why Computing Requires Symbols", *Proceedings of the Sixth Annual Conference of the Cognitive Science Society*, Boulder, Colorado, pp.71–73.

[146] Reddy, Michael (1979) "The Conduit Metaphor—A Case of Frame Conflict in our Language about Language", In: Ortony, Andrew (ed.), *Metaphor and Thought*, Cambridge University Press, Cambridge, England.

[147] Rickheit, Gert, Schnotz, Wolfgang, and Strohner, Hans (1985) "The Concept of Inference in Discourse Comprehension", In: *Inferences in Text Processing*, Rickheit G. and Strohner H. (eds.), Elsevier Science Publishers (North Holland), Amsterdam.

[148] Rieger, Charles (1975) "Conceptual Memory and Inference", In: Schank, Roger (ed.), *Conceptual Information Processing*, North-Holland, Amsterdam.

[149] Riesbeck, Christopher and Martin, Charles (1985) "Direct Memory Access Parsing", Technical Report no.354, Department of Computer Science, Yale University, New Haven, CT.

[150] Rosch, Eleanor (1974) "Linguistic Relativity", In: *Human Communication*, Silverstein, A. (ed.), Hasted Press, New York.

[151] Rumelhart, David E. (1975) "Notes on a Schema for Stories.", In: Bobrow, D.G. and Collins, A. (eds.), *Representation and Understanding: Studies in Cognitive Science*, Academic Press, New York.

[152] Rumelhart, David E. (1984) "The Emergence of Cognitive Phenomena from Subsymbolic Processes", *Proceedings of the Sixth Annual Conference of the Cognitive Science Society*, Boulder, Colorado, pp.59–62.

[153] Saussure, Ferdinand de (1916) *Cours de Linguistique*, translated by W. Baskin, Philosophical Library, New York, 1959.

[154] Schank, Roger (1972) "Conceptual Dependency: A Theory of Natural Language Understanding", *Cognitive Psychology*, pp.552–631.

[155] Schank, Roger (1982) *Dynamic Memory*, Cambridge University Press, New York.S

[156] Schank, Roger and Abelson, Robert (1977) *Scripts, Plans, Goals, and Understanding*, Lawrence Erlbaum Associates, Hillsdale, NJ.

[157] Schank, R., Collins, C.G., Davis, E., Johnson, P.N., Lytinen, S. and Reiser B.J. (1982) "What's the Point?", *Cognitive Science*, 6:255–276.

[158] Schank, Roger and Birnbaum, Lawrence (1984) "Memory, Meaning, and Syntax", In: Bever, Thomas, Carroll, John and Miller, Lance (eds.), *Talking Minds*, MIT Press, Cambridge, MA.

[159] Schnotz, Wolfgang (1985) "Selectivity in Drawing Inferences", In: *Inferences in Text Processing*, Rickheit G. and Strohner H. (eds.), Elsevier Science Publishers (North Holland), Amsterdam.

[160] Schubert, Lenhart (1986) "Are There Preference Trade-offs in Attachment Decisions?", *Proceedings of the National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, Philadelphia, PA, pp.601–605.

[161] Schweickert, Richard and Boruff, Brian (1986) "Short-Term Memory Capacity: Magic Number or Magic Spell?", *Journal of experimental Psychology: Learning, Memory, and Cognition*, 3:419–425.

[162] Searle, John (1979) "Literal Meaning", In: *Expression and Meaning*, Cambridge University Press, New York.

[163] Searle, John (1984) *Minds, Brains, and Science*. Harvard University Press, Cambridge, MA.

[164] Selman, Bart (1985) *Rule-based Processing in a Connectionist System for Natural Language Understanding*, Technical Report CSRI-168, Department of Computer Science, University of Toronto, Toronto. A summary of this work can be found in: Selman, Bart and Hirst, Graeme (1985) "A Rule-Based Connectionist Parsing System", *Proceedings of the Seventh Annual Conference of the Cognitive Science Society*, Irvine, California, pp.212–219.

[165] Selman, Bart and Hirst, Graeme (1987) "Parsing as an Energy Minimization Problem", to appear in: Davis, David (ed.), *Genetic Algorithms and Simulated Annealing*, Pitman Research Notes in Artificial Intelligence.

[166] Small, Steven L. (1980) *Word Expert Parsing: A Theory of Distributed Word-Based Natural Language Understanding*, Ph.D. Thesis, Technical Report 954, Department of Computer Science, University of Maryland.

[167] Small, Steven L. (1983) "Parsing as Cooperative Distributed Inference: Understanding through Memory Interactions", In: King, Margaret (ed.) *Parsing Natural Language*, Academic Press, London, England.

[168] Small, Steven and Rieger, Charles (1982) "Parsing and Comprehending with Word Experts", *Proceedings of the National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, Pittsburgh, PA, pp.247–250.

[169] Sowa, John (1984) *Conceptual Structures: Information Processing in Mind and Machine*, Addison-Wesley Publishing Company, Reading, MA.

[170] Spiro, Rand (1980) "Prior Knowledge and Story Processing", *Poetics*, 9:313–327.

[171] Squire, Larry (1987) *Memory and Brain*, Oxford University Press, New York.

[172] Stallard, David (1987) "The Logical Analysis of Lexical Ambiguity", *Proceedings of the 25th Annual Meeting of the ACL*, Stanford, CA, pp.179–185.

[173] Stevenson, Rosemary (1986) "The Time Course of Pronoun Comprehension", *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, Amherst, Massachusetts, pp.102–109.

[174] Thagard, Paul (1986) "Parallel Computation and the Mind-Body Problem", *Cognitive Science*, 10:301-318.

[175] Thorndike, Perry and Yekovich, Frank (1980) "A Critique of Schema-Based Theories of Human Story Memory", *Poetics*, 9:23–49.

[176] Tulving, Endel (1983) *Elements of Episodic Memory*, Oxford University Press, New York.

[177] Tulving, Endel (1984) "Précis of *Elements of Episodic Memory*", *The Behavioral and Brain Sciences*, 7:223-268.

[178] Vipond, Douglas and Hunt, Russell (1984) "Point-Driven Understanding: Pragmatic and Cognitive Dimensions of Literary Reading", *Poetics*, 13:261–277.

[179] Wagenaar, Willem (1988) "Calibration and the Effects of Knowledge and Reconstruction in Retrieval from Memory", *Cognition*, 28:277–296.

[180] Waltz, David and Pollack, Jordan (1985) "Massively Parallel Parsing: A Strongly Interactive Model of Natural Language Interpretation", *Cognitive Science*, 9:51–74.

[181] Whorf, Benjamin (1956) *Language, Thought, and Reality*, MIT Press, Cambridge, MA.

[182] Wilensky, Robert (1978) *Understanding Goal-Based Stories*, Research Report, Department of Computer Science, Yale University, New Haven, CT.

[183] Wilensky, Robert (1982) "Points: A Theory of the Structure of Stories in Memory", In: Lehnert, W. and Ringle, M. H. (eds.), *Strategies for Natural Language Processing*, Lawrence Erlbaum Associates, Hillsdale, NJ.

[184] Wilensky, Robert (1983a) "Story Grammars versus Story Points", *The Behavioral and Brain Sciences*, 6:579–623.

[185] Wilensky, Robert (1983b) *Planning and Understanding*, Addison-Wesley, Reading, MA.

[186] Wilensky, Robert (1986) "Some Problems and Proposals for Knowledge Representation", Report No. UCB/Computer Science Dept.86/294, Computer Science Division, University of California at Berkeley.

[187] Wilks, Yorick A. (1975) "A Preferential Pattern-Seeking Semantics for Natural Language Inference", *Artificial Intelligence*, 6:53–74.

[188] Wilks, Yorick, Huang, Xuiming, and Fass, Dan (1985) "Syntax, Preference, and Right Attachment", *Proceedings of the Ninth Joint Conference on Artificial Intelligence*, Los Angeles, CA., pp.779–784.

[189] Winograd, Terry (1983) *Language as a Cognitive Process: Syntax*, Addison-Wesley, Reading, MA.

[190] Winograd, Terry and Flores, Carlos (1986) *Understanding Computers and Cognition: A New Foundation for Design*, Ablex Publishing Corporation, Norwood, NJ.

[191] Wittgenstein, Ludwig (1953) *Philosophical Investigations*, Basil Blackwell, Oxford.

[192] Zavarin, Valentina (1983) "Stratification in Story", *Proceedings of the Fifth Annual Conference of the Cognitive Science Society*, Rochester, New York.

Appendix A

A Complete Trace

The trace for the sentence "John drinks gin" is provided as is, that is, in its most verbose and unedited form, in the next pages. Studying this trace, the reader will realize that several minor complexities of **IDIoT** have not been addressed in this dissertation. Queries about the exact *modus operandi* of the current prototype, which is constantly evolving, should be mailed to the author. The phrase 'manager tries dynamic construction of.... ' means that the manager is attempting to execute the expansion procedure of a buildable KU whose race to build has not yet expired but whose expansion procedure has not yet succeeded. If such an attempt succeeds, the trace will have the phrase 'which succeeds' immediately following.

```
starting a reading
candidacy length set to 200
expectation length set to 100
submission length set to 50
decay factor set to 0
letter quantum set to 10
detection threshold set to 1
STM capacity set to 50
Working Memory capacity set to 7
forcedDetectionSignal is -1
confirmationSignal is -2
expectationSignal is -3
inhibitionSignal is -4
presenceSignal is 1
reinforcementSignal is -5
submissionSignal is -6


innate MU startOfSentence read in at 0
  is recognized
  manager sends presenceSignal from startOfSentence to s-Start arriving at 1
  manager sends presenceSignal from startOfSentence to endOfSentence arriving at 1
  manager sends presenceSignal from startOfSentence to imperative-Start arriving at 1
MU imperative-Start in mode idle works at 1
  receives presenceSignal on startOfSentence
MU s-Start in mode idle works at 1
  receives presenceSignal on startOfSentence
MU endOfSentence in mode idle works at 1
  receives presenceSignal on startOfSentence
innate MU 'John' read in at 3
  is recognized
  manager expands
  manager sends forcedDetectionSignal from manager to singular  arriving at 3.01
  manager sends forcedDetectionSignal from manager to proper  arriving at 3.01
  manager sends forcedDetectionSignal from manager to person  arriving at 3.01
  manager sends forcedDetectionSignal from manager to male  arriving at 3.01
MU singular in mode idle works at 3.01
  becomes detected at: 3.01
MU proper in mode idle works at 3.01
  becomes detected at: 3.01
  manager sends presenceSignal from proper to NP arriving at 4.01
  manager sends presenceSignal from proper to lookForReference arriving at 5.01
  manager sends presenceSignal from proper to resetLookForReference arriving at 4.01
MU person in mode idle works at 3.01
  becomes detected at: 3.01
  manager expands
  manager sends forcedDetectionSignal from manager to rational  arriving at 3.02
  manager sends forcedDetectionSignal from manager to animate  arriving at 3.02
MU male in mode idle works at 3.01
  becomes detected at: 3.01
MU rational in mode idle works at 3.02
  becomes detected at: 3.02
MU animate in mode idle works at 3.02
  becomes detected at: 3.02
MU NP in mode idle works at 4.01
  receives presenceSignal on proper
  start candidacy race ending at 204.01 for constraint proper
  satisfaction of proper against threshold 1
  becomes detected at: 4.01
  manager expands
  manager sends forcedDetectionSignal from manager to NPhead  arriving at 4.02
  manager sends forcedDetectionSignal from manager to NPquant  arriving at 4.02
  manager sends forcedDetectionSignal from manager to NPmods  arriving at 4.02
  manager sends forcedDetectionSignal from manager to 3rdPerson  arriving at 4.02
  manager sends inhibitionSignal from manager to referredNP  arriving at 4.02
  manager sends presenceSignal from NP to directObject arriving at 5.01
  manager sends presenceSignal from NP to coi arriving at 5.01
  manager sends presenceSignal from NP to PP arriving at 5.01
  manager sends presenceSignal from NP to s-Start arriving at 5.01
  manager sends presenceSignal from NP to detDisagreement arriving at 5.01
  manager sends presenceSignal from NP to detAttach arriving at 5.01
  manager sends presenceSignal from NP to adjAttach arriving at 5.01
  manager sends presenceSignal from NP to forced-NP-PP-Attach arriving at 5.01
  manager sends presenceSignal from NP to referenceToPersonalPronoun arriving at 5.01
  manager sends presenceSignal from NP to lookForReference arriving at 5.01
  manager sends presenceSignal from NP to NP-PP-locationAttach arriving at 5.01
  manager sends presenceSignal from NP to NP-PP-timeAttach arriving at 5.01
MU resetLookForReference in mode idle works at 4.01
  receives presenceSignal on proper
  start candidacy race ending at 204.01 for constraint proper
  satisfaction of proper against threshold 1
  becomes detected at: 4.01
  manager sends inhibitionSignal from manager to lookForReference  arriving at 4.02
  manager sends inhibitionSignal from manager to referredNP  arriving at 4.02
MU lookForReference in mode idle works at 4.02
  is inhibited at 4.02
MU referredNP in mode idle works at 4.02
  is inhibited at 4.02
MU NPhead in mode idle works at 4.02
  becomes detected at: 4.02
MU NPquant in mode idle works at 4.02
  becomes detected at: 4.02
MU 3rdPerson in mode idle works at 4.02
  becomes detected at: 4.02
MU NPmods in mode idle works at 4.02
  becomes detected at: 4.02
MU PP in mode idle works at 5.01
  receives presenceSignal on NP
MU adjAttach in mode idle works at 5.01
  receives presenceSignal on NP
MU NP-PP-locationAttach in mode idle works at 5.01
  receives presenceSignal on NP
MU lookForReference in mode idle works at 5.01
  receives presenceSignal on NP
  receives presenceSignal on proper
  start candidacy race ending at 205.01 for constraint proper
  satisfaction of proper against threshold 1
  becomes detected at: 5.01
  manager sends presenceSignal from lookForReference to referredNP arriving at 6.01
MU forced-NP-PP-Attach in mode idle works at 5.01
  receives presenceSignal on NP
MU detDisagreement in mode idle works at 5.01
  receives presenceSignal on NP
  start candidacy race ending at 205.01 for constraint c1
  manager tries dynamic construction of detDisagreement at 5.01
MU s-Start in mode idle works at 5.01
  receives presenceSignal on NP
  start candidacy race ending at 205.01 for constraint NP
  satisfaction of NP against threshold 1
  becomes detected at: 5.01
  manager expands
  manager sends forcedDetectionSignal from manager to topNP  arriving at 5.02
  manager sends forcedDetectionSignal from manager to subject  arriving at 5.02
  manager tries dynamic construction of detDisagreement at 5.01
  manager sends inhibitionSignal from manager to imperative-Start  arriving at 5.02
  manager sends presenceSignal from s-Start to subj-verb-rel arriving at 6.01
MU coi in mode idle works at 5.01
  receives presenceSignal on NP
MU NP-PP-timeAttach in mode idle works at 5.01
  receives presenceSignal on NP
MU directObject in mode idle works at 5.01
  receives presenceSignal on NP
MU detAttach in mode idle works at 5.01
  receives presenceSignal on NP
MU referenceToPersonalPronoun in mode idle works at 5.01
```

```
  receives presenceSignal on NP
MU imperative-Start in mode idle works at 5.02
  is inhibited at 5.02
MU topNP in mode idle works at 5.02
  becomes detected at: 5.02
MU subject in mode idle works at 5.02
  becomes detected at: 5.02
MU subj-verb-rel in mode idle works at 6.01
  receives presenceSignal on s-Start
MU referredNP in mode idle works at 6.01
  receives presenceSignal on lookForReference
  start candidacy race ending at 206.01 for constraint lookForReference
  manager tries dynamic construction of referredNP at 6.01
innate MU 'drinks' read in at 43
  is recognized
  manager expands
  manager sends forcedDetectionSignal from manager to 3rdPersonSingForm  arriving at 43.01
  manager sends presenceSignal from 'drinks' to actionDrink arriving at 44
MU 3rdPersonSingForm in mode idle works at 43.01
  becomes detected at: 43.01
MU actionDrink in mode idle works at 44
  receives presenceSignal on 'drinks'
  start candidacy race ending at 244 for constraint 'drinks'
  satisfaction of 'drinks' against threshold 1
    for verb value is 0
  manager sends submissionSignal from actionDrink to verb arriving at 45
  manager sends submissionSignal from actionDrink to drinkBeverage arriving at 45
MU drinkBeverage in mode idle works at 45
  receives submissionSignal on actionDrink
MU verb in mode idle works at 45
  receives submissionSignal on actionDrink
  manager sends submissionSignal from verb to VP arriving at 46
MU VP in mode idle works at 46
  receives submissionSignal on verb
  in submission, a triggered constraint is verb
  satisfaction of verb against threshold 1
  manager sends submissionSignal from VP to endOfSentence arriving at 47
  manager sends submissionSignal from VP to subj-verb-rel arriving at 47
  manager sends submissionSignal from VP to newFact arriving at 47
  manager sends submissionSignal from VP to adverbAfterVerb arriving at 47
  manager sends submissionSignal from VP to imperative-Start arriving at 47
  manager sends submissionSignal from VP to actionObserve arriving at 47
  manager sends submissionSignal from VP to VP-PP-timeAttach arriving at 47
  manager sends submissionSignal from VP to NP-PP-timeAttach arriving at 47
  manager sends submissionSignal from VP to forced-VP-PP-Attach arriving at 47
  manager sends submissionSignal from VP to NP-PP-locationAttach arriving at 47
  manager sends submissionSignal from VP to VP-PP-attributeAttach arriving at 47
  manager sends submissionSignal from VP to VP-PP-instrumentAttach arriving at 47
MU endOfSentence in mode idle works at 47
  receives submissionSignal on VP
MU subj-verb-rel in mode idle works at 47
  receives submissionSignal on VP
  in submission, a triggered constraint is NP
  satisfaction of NP against threshold 1
  manager sends confirmationSignal from subj-verb-rel to VP arriving at 48
MU actionObserve in mode idle works at 47
  receives submissionSignal on VP
MU NP-PP-locationAttach in mode idle works at 47
  receives submissionSignal on VP
MU VP-PP-instrumentAttach in mode idle works at 47
  receives submissionSignal on VP
MU newFact in mode idle works at 47
  receives submissionSignal on VP
MU forced-VP-PP-Attach in mode idle works at 47
  receives submissionSignal on VP
MU adverbAfterVerb in mode idle works at 47
  receives submissionSignal on VP
MU imperative-Start in mode idle works at 47
```

```
  receives submissionSignal on VP
MU VP-PP-timeAttach in mode idle works at 47
  receives submissionSignal on VP
MU NP-PP-timeAttach in mode idle works at 47
  receives submissionSignal on VP
MU VP-PP-attributeAttach in mode idle works at 47
  receives submissionSignal on VP
MU VP in mode needAConfirmationBeforeConfirming works at 48
  receives confirmationSignal on subj-verb-rel
  confirmers are subj-verb-rel
  manager sends confirmationSignal from VP to verb arriving at 49
MU verb in mode toBeConfirmed works at 49
  receives confirmationSignal on VP
  confirmers are VP
  manager sends confirmationSignal from verb to actionDrink arriving at 50
MU actionDrink in mode immediateCandidate works at 50
  receives confirmationSignal on verb
  confirmers are verb
  manager sends reinforcementSignal from actionDrink to verb arriving at 51
  becomes detected at: 50
  forgets end of race deadline
  manager expands
  manager tries dynamic construction of referredNP at 50
  manager tries dynamic construction of detDisagreement at 50
  manager sends presenceSignal from actionDrink to drinkBeverage arriving at 51
MU drinkBeverage in mode idle works at 51
  receives presenceSignal on actionDrink
MU verb in mode toBeReinforced works at 51
  receives reinforcementSignal on actionDrink
  manager sends reinforcementSignal from verb to VP arriving at 52
  becomes detected at: 51
MU VP in mode toBeReinforced works at 52
  receives reinforcementSignal on verb
  manager sends reinforcementSignal from VP to subj-verb-rel arriving at 53
  becomes detected at: 52
  manager expands
  manager sends forcedDetectionSignal from manager to activeMood  arriving at 52.01
  manager sends forcedDetectionSignal from manager to mainVerb  arriving at 52.01
  manager sends forcedDetectionSignal from manager to VPmods  arriving at 52.01
  manager tries dynamic construction of referredNP at 52
  manager tries dynamic construction of detDisagreement at 52
  manager sends inhibitionSignal from manager to forced-NP-PP-Attach  arriving at 52.01
  manager sends presenceSignal from VP to imperative-Start arriving at 53
  manager sends presenceSignal from VP to newFact arriving at 53
  manager sends presenceSignal from VP to endOfSentence arriving at 53
  manager sends presenceSignal from VP to actionObserve arriving at 53
  manager sends presenceSignal from VP to VP-PP-attributeAttach arriving at 53
  manager sends presenceSignal from VP to VP-PP-instrumentAttach arriving at 53
  manager sends presenceSignal from VP to VP-PP-timeAttach arriving at 53
  manager sends presenceSignal from VP to forced-VP-PP-Attach arriving at 53
  manager sends presenceSignal from VP to NP-PP-locationAttach arriving at 53
  manager sends presenceSignal from VP to NP-PP-timeAttach arriving at 53
  manager sends presenceSignal from VP to adverbAfterVerb arriving at 53
MU forced-NP-PP-Attach in mode idle works at 52.01
  is inhibited at 52.01
MU VPmods in mode idle works at 52.01
  becomes detected at: 52.01
MU activeMood in mode idle works at 52.01
  becomes detected at: 52.01
MU mainVerb in mode idle works at 52.01
  becomes detected at: 52.01
  manager sends presenceSignal from mainVerb to endOfSentence arriving at 53.01
MU VP-PP-attributeAttach in mode idle works at 53
  receives presenceSignal on VP
MU subj-verb-rel in mode toBeReinforced works at 53
  receives reinforcementSignal on VP
  becomes detected at: 53
  manager expands
  manager sends forcedDetectionSignal from manager to clause  arriving at 53.01
  manager tries dynamic construction of referredNP at 53
  manager tries dynamic construction of detDisagreement at 53
  manager sends presenceSignal from subj-verb-rel to directObject arriving at 54
  manager sends presenceSignal from subj-verb-rel to newFact arriving at 54
  manager sends presenceSignal from subj-verb-rel to verb-complement-attachment arriving at 54
  manager sends presenceSignal from subj-verb-rel to sva-1 arriving at 54
  manager sends presenceSignal from subj-verb-rel to sva-2 arriving at 54
MU actionObserve in mode idle works at 53
  receives presenceSignal on VP
MU VP-PP-instrumentAttach in mode idle works at 53
  receives presenceSignal on VP
MU NP-PP-locationAttach in mode idle works at 53
  receives presenceSignal on VP
MU newFact in mode idle works at 53
  receives presenceSignal on VP
MU forced-VP-PP-Attach in mode idle works at 53
  receives presenceSignal on VP
MU adverbAfterVerb in mode idle works at 53
  receives presenceSignal on VP
MU imperative-Start in mode idle works at 53
  receives presenceSignal on VP
MU VP-PP-timeAttach in mode idle works at 53
  receives presenceSignal on VP
MU NP-PP-timeAttach in mode idle works at 53
  receives presenceSignal on VP
MU endOfSentence in mode idle works at 53
  receives presenceSignal on VP
MU endOfSentence in mode idle works at 53.01
  receives presenceSignal on mainVerb
MU clause in mode idle works at 53.01
  becomes detected at: 53.01
MU sva-2 in mode idle works at 54
  receives presenceSignal on subj-verb-rel
  start candidacy race ending at 254 for constraint subj-verb-rel
  manager tries dynamic construction of sva-2 at 54
MU newFact in mode idle works at 54
  receives presenceSignal on subj-verb-rel
MU verb-complement-attachment in mode idle works at 54
  receives presenceSignal on subj-verb-rel
MU directObject in mode idle works at 54
  receives presenceSignal on subj-verb-rel
MU sva-1 in mode idle works at 54
  receives presenceSignal on subj-verb-rel
  start candidacy race ending at 254 for constraint subj-verb-rel
  manager tries dynamic construction of sva-1 at 54
innate MU 'gin' read in at 103
  is recognized
  manager expands
  manager sends forcedDetectionSignal from manager to singular  arriving at 103.01
  manager sends presenceSignal from 'gin' to gin arriving at 104
MU singular in mode idle works at 103.01
  becomes detected at: 103.01
MU gin in mode idle works at 104
  receives presenceSignal on 'gin'
  start candidacy race ending at 304 for constraint 'gin'
  satisfaction of 'gin' against threshold 1
    for noun value is 0
  manager sends submissionSignal from gin to noun arriving at 105
MU noun in mode idle works at 105
  receives submissionSignal on gin
  manager sends submissionSignal from noun to NP arriving at 106
MU NP in mode idle works at 106
  receives submissionSignal on noun
  in submission, a triggered constraint is justNoun
  satisfaction of justNoun against threshold 1
  manager sends submissionSignal from NP to directObject arriving at 107
  manager sends submissionSignal from NP to coi arriving at 107
  manager sends submissionSignal from NP to s-Start arriving at 107
  manager sends submissionSignal from NP to detAttach arriving at 107
  manager sends submissionSignal from NP to referenceToPersonalPronoun arriving at 107
  manager sends submissionSignal from NP to PP arriving at 107
  manager sends submissionSignal from NP to lookForReference arriving at 107
  manager sends submissionSignal from NP to NP-PP-timeAttach arriving at 107
  manager sends submissionSignal from NP to forced-NP-PP-Attach arriving at 107
  manager sends submissionSignal from NP to NP-PP-locationAttach arriving at 107
  manager sends submissionSignal from NP to adjAttach arriving at 107
  manager sends submissionSignal from NP to detDisagreement arriving at 107
MU PP in mode idle works at 107
  receives submissionSignal on NP
MU adjAttach in mode idle works at 107
  receives submissionSignal on NP
MU NP-PP-locationAttach in mode idle works at 107
  receives submissionSignal on NP
MU lookForReference in mode idle works at 107
  receives submissionSignal on NP
MU forced-NP-PP-Attach in mode idle works at 107
  receives submissionSignal on NP
MU detDisagreement in mode toBeBuiltImmediate works at 107
  receives submissionSignal on NP
MU s-Start in mode idle works at 107
  receives submissionSignal on NP
MU coi in mode idle works at 107
  receives submissionSignal on NP
MU NP-PP-timeAttach in mode idle works at 107
  receives submissionSignal on NP
MU directObject in mode idle works at 107
  receives submissionSignal on NP
  in submission, a triggered constraint is c1
  satisfaction of c1 against threshold 1
  manager attempts dynamic construction for submission at 107
  manager sends confirmationSignal from directObject to NP arriving at 108.26
MU detAttach in mode idle works at 107
  receives submissionSignal on NP
MU referenceToPersonalPronoun in mode idle works at 107
  receives submissionSignal on NP
MU NP in mode needAConfirmationBeforeConfirming works at 108.26
  receives confirmationSignal on directObject
  confirmers are directObject
  manager sends confirmationSignal from NP to noun arriving at 109.26
MU noun in mode toBeConfirmed works at 109.26
  receives confirmationSignal on NP
  confirmers are NP
  manager sends confirmationSignal from noun to gin arriving at 110.26
MU gin in mode immediateCandidate works at 110.26
  receives confirmationSignal on noun
  confirmers are noun
  manager sends reinforcementSignal from gin to noun arriving at 111.26
  becomes detected at: 110.26
  forgets end of race deadline
  manager expands
  manager tries dynamic construction of sva-1 at 110.26
  manager tries dynamic construction of sva-2 at 110.26
  manager tries dynamic construction of referredNP at 110.26
  manager tries dynamic construction of detDisagreement at 110.26
  manager sends confirmationSignal from gin to cardGameGin arriving at 111.26
  manager sends confirmationSignal from gin to alcoholicBeverageGin arriving at 111.26
MU alcoholicBeverageGin in mode idle works at 111.26
  receives confirmationSignal on gin
  start direct candidacy race ending at 311.26 for constraint gin
  manager sends submissionSignal from alcoholicBeverageGin to alcoholicBeverage arriving at 112.26
MU cardGameGin in mode idle works at 111.26
  receives confirmationSignal on gin
  start direct candidacy race ending at 311.26 for constraint gin
  manager sends submissionSignal from cardGameGin to cardGame arriving at 112.26
  manager sends submissionSignal from cardGameGin to rulesOfGin arriving at 112.26
```

```
MU noun in mode toBeReinforced works at 111.26
  receives reinforcementSignal on gin
  manager sends reinforcementSignal from noun to NP arriving at 112.26
  becomes detected at: 111.26
MU alcoholicBeverage in mode idle works at 112.26
  receives submissionSignal on alcoholicBeverageGin
  in submission, a triggered constraint is alcoholicBeverageGin
  satisfaction of alcoholicBeverageGin against threshold 1
  manager sends submissionSignal from alcoholicBeverage to aboutAlcoholBeverages arriving at 113.26
  manager sends submissionSignal from alcoholicBeverage to beverage arriving at 113.26
MU NP in mode toBeReinforced works at 112.26
  receives reinforcementSignal on noun
  manager sends reinforcementSignal from NP to directObject arriving at 113.26
  becomes detected at: 112.26
  manager expands
  manager sends forcedDetectionSignal from manager to NPhead  arriving at 112.27
  manager sends forcedDetectionSignal from manager to NPquant  arriving at 112.27
  manager sends forcedDetectionSignal from manager to NPmods  arriving at 112.27
  manager sends forcedDetectionSignal from manager to 3rdPerson  arriving at 112.27
  manager tries dynamic construction of sva-1 at 112.26
  manager tries dynamic construction of sva-2 at 112.26
  manager tries dynamic construction of referredNP at 112.26
  manager tries dynamic construction of detDisagreement at 112.26
  manager sends inhibitionSignal from manager to referredNP  arriving at 112.27
  manager sends presenceSignal from NP to coi arriving at 113.26
  manager sends presenceSignal from NP to PP arriving at 113.26
  manager sends presenceSignal from NP to s-Start arriving at 113.26
  manager sends presenceSignal from NP to detDisagreement arriving at 113.26
  manager sends presenceSignal from NP to detAttach arriving at 113.26
  manager sends presenceSignal from NP to adjAttach arriving at 113.26
  manager sends presenceSignal from NP to forced-NP-PP-Attach arriving at 113.26
  manager sends presenceSignal from NP to referenceToPersonalPronoun arriving at 113.26
  manager sends presenceSignal from NP to lookForReference arriving at 113.26
  manager sends presenceSignal from NP to NP-PP-locationAttach arriving at 113.26
  manager sends presenceSignal from NP to NP-PP-timeAttach arriving at 113.26
MU cardGame in mode idle works at 112.26
  receives submissionSignal on cardGameGin
  in submission, a triggered constraint is cardGameGin
  satisfaction of cardGameGin against threshold 1
  manager sends submissionSignal from cardGame to game arriving at 113.26
  manager sends submissionSignal from cardGame to aboutPlayingCards arriving at 113.26
MU rulesOfGin in mode idle works at 112.26
  receives submissionSignal on cardGameGin
  in submission, a triggered constraint is cardGameGin
  satisfaction of cardGameGin against threshold 1
MU NPquant in mode idle works at 112.27
  becomes detected at: 112.27
MU 3rdPerson in mode idle works at 112.27
  becomes detected at: 112.27
MU referredNP in mode toBeBuiltImmediate works at 112.27
  is inhibited at 112.27
  forgets end of race deadline
MU NPmods in mode idle works at 112.27
  becomes detected at: 112.27
MU NPhead in mode idle works at 112.27
  becomes detected at: 112.27
MU PP in mode idle works at 113.26
  receives presenceSignal on NP
MU game in mode idle works at 113.26
  receives submissionSignal on cardGame
  in submission, a triggered constraint is cardGame
  satisfaction of cardGame against threshold 1
  manager sends submissionSignal from game to actionPlayAGame arriving at 114.26
MU beverage in mode idle works at 113.26
  receives submissionSignal on alcoholicBeverage
  in submission, a triggered constraint is alcoholicBeverage
  satisfaction of alcoholicBeverage against threshold 1
  manager sends submissionSignal from beverage to drinkBeverage arriving at 114.26
  manager sends submissionSignal from beverage to liquid arriving at 114.26
MU aboutAlcoholBeverages in mode idle works at 113.26
  receives submissionSignal on alcoholicBeverage
  in submission, a triggered constraint is c1
  satisfaction of c1 against threshold 1
MU adjAttach in mode idle works at 113.26
  receives presenceSignal on NP
MU NP-PP-locationAttach in mode idle works at 113.26
  receives presenceSignal on NP
MU forced-NP-PP-Attach in mode idle works at 113.26
  receives presenceSignal on NP
MU lookForReference in mode idle works at 113.26
  receives presenceSignal on NP
MU detDisagreement in mode toBeBuiltImmediate works at 113.26
  receives presenceSignal on NP
MU s-Start in mode idle works at 113.26
  receives presenceSignal on NP
MU aboutPlayingCards in mode idle works at 113.26
  receives submissionSignal on cardGame
  in submission, a triggered constraint is c1
  satisfaction of c1 against threshold 1
MU coi in mode idle works at 113.26
  receives presenceSignal on NP
MU NP-PP-timeAttach in mode idle works at 113.26
  receives presenceSignal on NP
MU directObject in mode toBeReinforced works at 113.26
  receives reinforcementSignal on NP
  becomes detected at: 113.26
  manager expands
  manager sends forcedDetectionSignal from manager to verbComplemented  arriving at 113.27
  manager sends forcedDetectionSignal from manager to topNP  arriving at 113.27
  manager tries dynamic construction of sva-1 at 113.26
  manager tries dynamic construction of sva-2 at 113.26
  manager tries dynamic construction of detDisagreement at 113.26
  manager sends presenceSignal from directObject to coi arriving at 114.26
  manager sends presenceSignal from directObject to newFact arriving at 114.26
  manager sends presenceSignal from directObject to beNP arriving at 114.26
MU detAttach in mode idle works at 113.26
  receives presenceSignal on NP
MU referenceToPersonalPronoun in mode idle works at 113.26
  receives presenceSignal on NP
MU verbComplemented in mode idle works at 113.27
  becomes detected at: 113.27
MU topNP in mode idle works at 113.27
  becomes detected at: 113.27
MU newFact in mode idle works at 114.26
  receives presenceSignal on directObject
MU drinkBeverage in mode idle works at 114.26
  receives submissionSignal on beverage
  in submission, a triggered constraint is c1
  satisfaction of c1 against threshold 1
  manager attempts dynamic construction for submission at 114.26
  manager sends confirmationSignal from drinkBeverage to beverage arriving at 115.52
MU liquid in mode idle works at 114.26
  receives submissionSignal on beverage
  in submission, a triggered constraint is beverage
  satisfaction of beverage against threshold 1
  manager sends submissionSignal from liquid to actionSpillLiquid arriving at 115.26
  manager sends submissionSignal from liquid to aboutLiquids arriving at 115.26
MU actionPlayAGame in mode idle works at 114.26
  receives submissionSignal on game
MU coi in mode idle works at 114.26
  receives presenceSignal on directObject
MU beNP in mode idle works at 114.26
  receives presenceSignal on directObject
MU actionSpillLiquid in mode idle works at 115.26
  receives submissionSignal on liquid
MU aboutLiquids in mode idle works at 115.26
```

```
  receives submissionSignal on liquid
  in submission, a triggered constraint is liquid
  satisfaction of liquid against threshold 1
MU beverage in mode needAConfirmationBeforeConfirming works at 115.52
  receives confirmationSignal on drinkBeverage
  confirmers are drinkBeverage
  manager sends confirmationSignal from beverage to alcoholicBeverage arriving at 115.53
MU alcoholicBeverage in mode needAConfirmationBeforeConfirming works at 115.53
  manager sends confirmationSignal from alcoholicBeverage to alcoholicBeverageGin arriving at 115.54
MU alcoholicBeverageGin in mode directImmediateCandidate works at 115.54
  becomes detected at: 115.54
  forgets end of race deadline
  manager expands
  manager sends forcedDetectionSignal from manager to tasteOfGin arriving at 115.55
  manager tries dynamic construction of sva-1 at 115.54
  manager tries dynamic construction of sva-2 at 115.54
  manager tries dynamic construction of detDisagreement at 115.54
  manager sends forcedDetectionSignal from alcoholicBeverageGin to alcoholicBeverage arriving at 115.55
MU tasteOfGin in mode idle works at 115.55
  becomes detected at: 115.55
MU alcoholicBeverage in mode toBeReinforced works at 115.55
  receives forcedDetectionSignal on alcoholicBeverageGin
  becomes detected at: 115.55
  manager expands
  manager tries dynamic construction of sva-1 at 115.55
  manager tries dynamic construction of sva-2 at 115.55
  manager tries dynamic construction of detDisagreement at 115.55
  manager sends forcedDetectionSignal from alcoholicBeverage
to aboutAlcoholBeverages beverage  arriving at 115.56
MU aboutAlcoholBeverages in mode needAConfirmationBeforeConfirming works at 115.56
  receives forcedDetectionSignal on alcoholicBeverage
  becomes detected at: 115.56
  manager expands
  manager tries dynamic construction of sva-1 at 115.56
  manager tries dynamic construction of sva-2 at 115.56
  manager tries dynamic construction of detDisagreement at 115.56
MU beverage in mode toBeReinforced works at 115.56
  receives forcedDetectionSignal on alcoholicBeverage
  becomes detected at: 115.56
  manager expands
  manager tries dynamic construction of sva-1 at 115.56
  manager tries dynamic construction of sva-2 at 115.56
  manager tries dynamic construction of detDisagreement at 115.56
  manager sends forcedDetectionSignal from beverage to liquid  arriving at 115.57
MU liquid in mode needAConfirmationBeforeConfirming works at 115.57
  receives forcedDetectionSignal on beverage
  becomes detected at: 115.57
  manager expands
  manager tries dynamic construction of sva-1 at 115.57
  manager tries dynamic construction of sva-2 at 115.57
  manager tries dynamic construction of detDisagreement at 115.57
  manager sends forcedDetectionSignal from liquid to aboutLiquids  arriving at 115.58
  manager sends presenceSignal from liquid to actionSpillLiquid arriving at 116.57
MU aboutLiquids in mode needAConfirmationBeforeConfirming works at 115.58
  receives forcedDetectionSignal on liquid
  becomes detected at: 115.58
  manager expands
  manager tries dynamic construction of sva-1 at 115.58
  manager tries dynamic construction of sva-2 at 115.58
  manager tries dynamic construction of detDisagreement at 115.58
MU actionSpillLiquid in mode idle works at 116.57
  receives presenceSignal on liquid
innate MU . read in at 133
  is recognized
  manager sends presenceSignal from . to endOfSentence arriving at 134
innate MU endOfText read in at 133.01
  is recognized
  manager sends presenceSignal from endOfText to endOfSentence arriving at 134.01
```

```
MU endOfSentence in mode idle works at 134
  receives presenceSignal on .
MU endOfSentence in mode idle works at 134.01
  receives presenceSignal on endOfText
  start candidacy race ending at 334.01 for constraint c2
  satisfaction of c2 against threshold 1
  becomes detected at: 134.01
  manager tries dynamic construction of sva-1 at 134.01
  manager tries dynamic construction of sva-2 at 134.01
  manager tries dynamic construction of detDisagreement at 134.01
  manager sends presenceSignal from endOfSentence to newFact arriving at 135.01
  manager sends presenceSignal from endOfSentence to missingDirectObject arriving at 135.01
  manager sends presenceSignal from endOfSentence to absentLocation arriving at 135.01
MU newFact in mode idle works at 135.01
  receives presenceSignal on endOfSentence
  start candidacy race ending at 335.01 for constraint subject-verb
  satisfaction of subject-verb against threshold 1
  becomes detected at: 135.01
  manager expands
  manager tries dynamic construction of sva-1 at 135.01
  manager tries dynamic construction of sva-2 at 135.01
  manager tries dynamic construction of detDisagreement at 135.01
  manager sends forcedDetectionSignal from newFact to removeTopNP resetFeatures  arriving at 135.02
  manager sends presenceSignal from newFact to complement arriving at 136.01
  manager sends presenceSignal from newFact to rel-clause arriving at 136.01
  manager sends presenceSignal from newFact to beAttribute arriving at 136.01
  manager sends presenceSignal from newFact to beNP arriving at 136.01
MU absentLocation in mode idle works at 135.01
  receives presenceSignal on endOfSentence
MU missingDirectObject in mode idle works at 135.01
  receives presenceSignal on endOfSentence
MU removeTopNP in mode idle works at 135.02
  receives forcedDetectionSignal on newFact
  becomes detected at: 135.02
  manager expands
  manager tries dynamic construction of sva-1 at 135.02
  manager tries dynamic construction of sva-2 at 135.02
  manager tries dynamic construction of detDisagreement at 135.02
MU resetFeatures in mode idle works at 135.02
  receives forcedDetectionSignal on newFact
  becomes detected at: 135.02
  manager tries dynamic construction of sva-1 at 135.02
  manager tries dynamic construction of sva-2 at 135.02
  manager tries dynamic construction of detDisagreement at 135.02
  manager sends inhibitionSignal from manager to sva-1  arriving at 135.03
  manager sends inhibitionSignal from manager to sva-2  arriving at 135.03
  manager sends inhibitionSignal from manager to referredNP  arriving at 135.03
  manager sends inhibitionSignal from manager to directObject  arriving at 135.03
  manager sends inhibitionSignal from manager to coi  arriving at 135.03
  manager sends inhibitionSignal from manager to missingDirectObject  arriving at 135.03
  manager sends inhibitionSignal from manager to absentLocation  arriving at 135.03
  manager sends inhibitionSignal from manager to detDisagreement  arriving at 135.03
  manager sends inhibitionSignal from manager to forced-NP-PP-Attach  arriving at 135.03
  manager sends inhibitionSignal from manager to forced-VP-PP-Attach  arriving at 135.03
  manager sends inhibitionSignal from manager to adjStart  arriving at 135.03
  manager sends inhibitionSignal from manager to NP  arriving at 135.03
  manager sends inhibitionSignal from manager to VP-PP-attributeAttach  arriving at 135.03
  manager sends inhibitionSignal from manager to VP-PP-instrumentAttach  arriving at 135.03
  manager sends inhibitionSignal from manager to VP-PP-locationAttach  arriving at 135.03
  manager sends inhibitionSignal from manager to VP-PP-timeAttach  arriving at 135.03
  manager sends inhibitionSignal from manager to beNP  arriving at 135.03
  manager sends inhibitionSignal from manager to bePP  arriving at 135.03
  manager sends inhibitionSignal from manager to adjAttach  arriving at 135.03
  manager sends inhibitionSignal from manager to atLocation  arriving at 135.03
  manager sends inhibitionSignal from manager to atTime  arriving at 135.03
  manager sends inhibitionSignal from manager to inLocation  arriving at 135.03
  manager sends inhibitionSignal from manager to inTime  arriving at 135.03
  manager sends inhibitionSignal from manager to onLocation  arriving at 135.03
  manager sends inhibitionSignal from manager to onTime  arriving at 135.03
  manager sends inhibitionSignal from manager to checkLocationForAttribute  arriving at 135.03
  manager sends inhibitionSignal from manager to detAttach  arriving at 135.03
  manager sends inhibitionSignal from manager to referenceToPersonalPronoun  arriving at 135.03
  manager sends inhibitionSignal from manager to rel-pronoun-disagreement  arriving at 135.03
  manager sends inhibitionSignal from manager to verb-complement-attachment  arriving at 135.03
  manager sends inhibitionSignal from manager to withAttribute  arriving at 135.03
  manager sends inhibitionSignal from manager to withInstrument  arriving at 135.03
  manager sends inhibitionSignal from manager to NP-PP-locationAttach  arriving at 135.03
  manager sends inhibitionSignal from manager to NP-PP-attributeAttach  arriving at 135.03
  manager sends inhibitionSignal from manager to NP-PP-timeAttach  arriving at 135.03
  manager sends inhibitionSignal from manager to referenceAttachment1  arriving at 135.03
  manager sends inhibitionSignal from manager to referenceAttachment2  arriving at 135.03
  manager sends inhibitionSignal from manager to cleanUpCompulsoryLocation  arriving at 135.03


MU cleanUpCompulsoryLocation in mode idle works at 135.03
  is inhibited at 135.03
MU absentLocation in mode idle works at 135.03
  is inhibited at 135.03
MU referredNP in mode idle works at 135.03
  is inhibited at 135.03
MU referenceAttachment2 in mode idle works at 135.03
  is inhibited at 135.03
MU adjAttach in mode idle works at 135.03
  is inhibited at 135.03
MU referenceAttachment1 in mode idle works at 135.03
  is inhibited at 135.03
MU bePP in mode idle works at 135.03
  is inhibited at 135.03
MU VP-PP-locationAttach in mode idle works at 135.03
  is inhibited at 135.03
MU NP-PP-attributeAttach in mode idle works at 135.03
  is inhibited at 135.03
MU VP-PP-instrumentAttach in mode idle works at 135.03
  is inhibited at 135.03
MU sva-2 in mode toBeBuiltImmediate works at 135.03
  is inhibited at 135.03
  forgets end of race deadline
MU forced-NP-PP-Attach in mode idle works at 135.03
  is inhibited at 135.03
MU inLocation in mode idle works at 135.03
  is inhibited at 135.03
MU checkLocationForAttribute in mode idle works at 135.03
  is inhibited at 135.03
MU forced-VP-PP-Attach in mode idle works at 135.03
  is inhibited at 135.03
MU withInstrument in mode idle works at 135.03
  is inhibited at 135.03
MU NP-PP-locationAttach in mode idle works at 135.03
  is inhibited at 135.03
MU coi in mode idle works at 135.03
  is inhibited at 135.03
MU VP-PP-timeAttach in mode idle works at 135.03
  is inhibited at 135.03
MU directObject in mode idle works at 135.03
  is inhibited at 135.03
MU NP-PP-timeAttach in mode idle works at 135.03
  is inhibited at 135.03
MU detAttach in mode idle works at 135.03
  is inhibited at 135.03
MU referenceToPersonalPronoun in mode idle works at 135.03
  is inhibited at 135.03
MU verb-complement-attachment in mode idle works at 135.03
  is inhibited at 135.03
MU rel-pronoun-disagreement in mode idle works at 135.03
  is inhibited at 135.03
MU missingDirectObject in mode idle works at 135.03
```

```
  is inhibited at 135.03
MU NP in mode idle works at 135.03
  is inhibited at 135.03
MU beNP in mode idle works at 135.03
  is inhibited at 135.03
MU onTime in mode idle works at 135.03
  is inhibited at 135.03
MU atTime in mode idle works at 135.03
  is inhibited at 135.03
MU sva-1 in mode toBeBuiltImmediate works at 135.03
  is inhibited at 135.03
  forgets end of race deadline
MU adjStart in mode idle works at 135.03
  is inhibited at 135.03
MU inTime in mode idle works at 135.03
  is inhibited at 135.03
MU detDisagreement in mode toBeBuiltImmediate works at 135.03
  is inhibited at 135.03
  forgets end of race deadline
MU withAttribute in mode idle works at 135.03
  is inhibited at 135.03
MU atLocation in mode idle works at 135.03
  is inhibited at 135.03
MU onLocation in mode idle works at 135.03
  is inhibited at 135.03
MU VP-PP-attributeAttach in mode idle works at 135.03
  is inhibited at 135.03
MU beAttribute in mode idle works at 136.01
  receives presenceSignal on newFact
MU beNP in mode idle works at 136.01
  receives presenceSignal on newFact
MU rel-clause in mode idle works at 136.01
  receives presenceSignal on newFact
MU complement in mode idle works at 136.01
  receives presenceSignal on newFact
MU cardGameGin in mode directImmediateCandidate works at 311.26
  reverts to idle
end of reading
```