IN-61-CR
43118
P-27

Machine Learning Research 1989-90: Final Report

Bruce W. Porter
Arthur Souther

Research Staff:
Kenneth Murray
James Lester
Liane Acker
Erik Eilerts
David Severinsen

University of Texas at Austin

June 1, 1990

Cooperative Agreement NCC 9-16
Research Activity No. ET.14

NASA Johnson Space Center
Mission Support Directorate

(NASA-CR-188828) MACHINE LEARNING RESEARCH 1989-90 Final Report (Research Inst. for Advanced Computer Science) 27 p CSCL 09B

N91-32826

Unclas
G3/61 0043118

Research Institute for Computing and Information Systems
University of Houston - Clear Lake

T·E·C·H·N·I·C·A·L R·E·P·O·R·T

The RICIS Concept

The University of Houston-Clear Lake established the Research Institute for Computing and Information systems in 1986 to encourage NASA Johnson Space Center and local industry to actively support research in the computing and information sciences. As part of this endeavor, UH-Clear Lake proposed a partnership with JSC to jointly define and manage an integrated program of research in advanced data processing technology needed for JSC's main missions, including administrative, engineering and science responsibilities. JSC agreed and entered into a three-year cooperative agreement with UH-Clear Lake beginning in May, 1986, to jointly plan and execute such research through RICIS. Additionally, under Cooperative Agreement NCC 9-16, computing and educational facilities are shared by the two institutions to conduct the research.

The mission of RICIS is to conduct, coordinate and disseminate research on computing and information systems among researchers, sponsors and users from UH-Clear Lake, NASA/JSC, and other research organizations. Within UH-Clear Lake, the mission is being implemented through interdisciplinary involvement of faculty and students from each of the four schools: Business, Education, Human Sciences and Humanities, and Natural and Applied Sciences.

Other research organizations are involved via the "gateway" concept. UH-Clear Lake establishes relationships with other universities and research organizations, having common research interests, to provide additional sources of expertise to conduct needed research.

A major role of RICIS is to find the best match of sponsors, researchers and research objectives to advance knowledge in the computing and information sciences. Working jointly with NASA/JSC, RICIS advises on research needs, recommends principals for conducting the research, provides technical and administrative support to coordinate the research, and integrates technical results into the cooperative goals of UH-Clear Lake and NASA/JSC.

Machine Learning Research 1989-90: Final Report

Preface

This research was conducted under auspices of the Research Institute for Computing and Information Systems by Bruce W. Porter and Arthur Souther of the University of Texas at Austin. Additional research staff included Kenneth Murray, James Lester, Liane Acker, Erik Eilerts and David Severinsen. Dr. Glenn Freedman served as RICIS technical representative.

Funding has been provided by the Mission Support Directorate, NASA/JSC through Cooperative Agreement NCC 9-16 between NASA Johnson Space Center and the University of Houston-Clear Lake. The NASA technical monitor for this activity was Robert B. MacDonald, Assistant for Research, Education & University Programs for the Mission Support Directorate, NASA/JSC.

The views and conclusions contained in this report are those of the authors and should not be interpreted as representative of the official policies, either express or implied, of NASA or the United States Government.

[illegible]

Machine Learning Research 1989-90: Final Report[1]

Bruce W. Porter   Arthur Souther

June 1, 1990

Research Staff:
Kenneth Murray
James Lester
Liane Acker
Erik Eilerts
David Severinsen

Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712
porter@cs.utexas.edu

[1]Support for this research was provided by the Air Force Human Resources Laboratory under RICIS grant ET.14.

Contents

1 Research Objectives and Accomplishments

The long-term goal of our research is to develop technology for constructing and using large-scale, multifunctional knowledge bases on computers. These knowledge bases would significantly improve current expert systems and tutoring systems because they contain the broad knowledge of a domain required to perform multiple tasks [1, 14, 12]. For example, a multifunctional knowledge base for a new aircraft might support expert programs for assembly, maintenance, instruction, and design modification.

Building a single knowledge base that supports multiple tasks has two significant advantages over building separate knowledge bases for each task. First, the effort of building a multifunctional knowledge base can be amortized over many expert system projects. Using existing technology (*e.g.*, [36, 5]), multifunctional knowledge bases can be compiled into efficient expert systems for performing disparate tasks within the domain. In contrast, reusing a knowledge base built for a single task is typically infeasible because the knowledge is overly specific. For example, Clancey [6] documents the difficulties in reusing the Mycin medical diagnosis knowledge base for tutoring. The second advantage of multifunctional knowledge bases is a significant reduction in the brittleness of expert systems. Multifunctional knowledge bases contain fundamental domain knowledge that can help solve problems that are beyond the range of task-specific expert systems. For example, Fink [8] uses fundamental knowledge of the structure and function of complex mechanisms to supplement surface-level heuristics for diagnosing faults. Applying the principle on a large scale, the CYC knowledge base is intended to provide a comprehensive body of task-independent knowledge "to provide assistance for expert systems, natural language understanders, and so on, as they get 'stuck' on problems" [15].

Unfortunately, multifunctional knowledge bases are hard to build with current methods for knowledge engineering and knowledge acquisition. These methods do not address the problems caused by the size and complexity of multifunctional knowledge bases. As a knowledge base grows, it becomes increasingly difficult to maintain, and determining the consequences of a change to the knowledge base becomes difficult and error-prone [33]. Numerous surveys of methods for building large knowledge bases (*e.g.*, [22, 30, 38]) identify these problems as serious obstacles to the advance of knowledge base technology.

Our research during the past twelve months has produced technology for building and using multifunctional knowledge bases. In particular, we have developed prototypes systems for the following:

- Knowledge engineering – This technology facilitates viewing and editing the contents of a large knowledge base.
- Knowledge acquisition – This technology integrates new information from a domain expert into a knowledge base by automatically determining its consequences and adapting the existing knowledge.

- Knowledge access – This technology accesses multifunctional knowledge bases to extract knowledge that coherently answers questions.

Continuing this research, we plan to significantly improve these prototype systems and to integrate them into a single framework for constructing and maintaining multifunctional knowledge bases.

2 Knowledge Engineering

We have developed a prototype knowledge-engineering environment for building multifunctional knowledge bases. This environment provides a language for representing knowledge and software support for viewing and editing knowledge structures. We describe each of these in turn.

2.1 Knowledge Representation

Our knowledge representation language shares the primary tenets of other modern languages, such as KnowledgeCraft [4], KEE [10], Strobe [32], and CYC [14]. These tenets include the following:

1. Declarative knowledge is represented with frames (or objects) and procedural knowledge is represented with rules. The results of every computation are cachable as declarations.

2. Constraints on knowledge base entries are explicitly represented and enforced by the language.

3. Commonly used inference methods, such as inheritance, are built into the language, and others can be defined by the user.

Our knowledge representation language builds on Theo, a language developed at Carnegie-Mellon University [23]. We have added methods for representing rules and constraints. Our remaining work is to develop inference methods such as inheritance and forward chaining.

In addition to this basic functionality, our representation language provides features important for building multifunctional knowledge bases. Of utmost importance is the ability to represent viewpoints, which are collections of facts that should be considered together. For example, the viewpoint "car as a manufactured artifact" contains information about raw materials and the assembly process, while the viewpoint "car as a consumer durable" contains information about purchase costs and longevity. A multifunctional knowledge base contains many, highly-integrated viewpoints for each concept.

Past research on using viewpoints for organizing knowledge has assumed that all viewpoints are represented explicitly. Viewpoints in Swartout's XPLAIN system [36] consist of annotations on elements of domain knowledge that indicate when a piece of knowledge should be included in an explanation. Viewpoints in McKeown's ADVISOR system [21] are represented by multiple hierarchies, each representing a single perspective. Viewpoints in McCoy's system [18] are represented by lists associated with each object in the knowledge base; each list specifies the salience of each of the object's properties under a particular viewpoint. Unfortunately, explicitly representing viewpoints for a large knowledge base is infeasible.

Our research addresses this problem with methods for creating viewpoints when they are needed [34, 2, 25]. As explained below, this is done using a relatively small number of general viewpoints, which we call "view types," that are instantiated for specific concepts.

2.2 Software for Viewing and Editing Knowledge Structures

We have developed prototype software for viewing and editing knowledge structures. Using mouse and menu operations, the knowledge engineer can "navigate" through a complex structure and selectively display it both graphically and textually. Numerous editing operations are available, such as adding an object to a graph, changing an object's attributes, and creating a rule to compute information when required.

This basic functionality is similar to that provided in other software environments for knowledge engineering (such as KEE [10], Strobe [31], and KnowledgeCraft [4]). However, we chose not to use commercial systems because an important goal of our research is to develop an integrated tool for knowledge engineering and knowledge acquisition. Because of the difficulties in extending commercial systems (*e.g.,* the unavailability of source code), we have replicated their functionality in our software.

We plan to significantly extend the basic functionality of this software. From three year's experience building a large knowledge base [29], we have found that graphical displays and graphical editing are very effective. Our domain experts use graphs to organize domain knowledge and to communicate with others. Once everyone agrees on a graph, our knowledge engineers convert it to the representation language. The software that we will add to our knowledge engineering environment will automate this conversion process, thereby allowing a domain expert to extend and modify the knowledge base by creating and editing graphs.

3 Knowledge Acquisition

The major obstacle to building multifunctional knowledge bases results from their size and complexity. Knowledge base modifications that are intended to correct one shortcoming may conflict with existing knowledge and introduce new problems. For example, extending a drug therapy advisor (*e.g.,* Mycin) to minimize the number of drugs prescribed to each patient conflicts with other therapy goals, such as maximizing the number of symptoms covered by the prescribed treatment [24]. Identifying how new information conflicts with existing knowledge is difficult: conflicts are often implicit, and the complexity of identifying interaction

Domain Expert: The leaf epidermis is covered by the leaf cuticle, which is composed of cutin.

KI: Cutin is impermeable to gases. Does the cuticle restrict water loss from the leaf?

Domain Expert: Yes, that's right.

KI: But the cuticle would also cause the leaf to starve.

Domain Expert: Explain.

KI: The cuticle is impermeable to gases. This prevents carbon dioxide in the atmosphere from passing through the leaf's epidermis. Without carbon dioxide, the leaf cannot conduct photosynthesis and starves.

Domain Expert: Well, cuticle is impermeable to carbon dioxide; however, the leaf survives.

KI: Does the cuticle only partially cover the epidermis? Or, perhaps there are portals in the epidermis that permit restricted gas flow?

Domain Expert: Yes, the epidermis does have portals. They're called stomata.

Figure 1: This figure illustrates the interaction between KI and a domain expert as new information describing Leaf Cuticle is integrated into the knowledge base. KI identifies unanticipated consequences of the new information that reveals a gap in the knowledge base. Then KI elicits additional knowledge to fill the gap.

between new information and existing knowledge increases with the size of the knowledge base. Developing the technology to determine how new information interacts with existing knowledge is the principle requirement for supporting the construction and maintenance of very large, multifunctional knowledge bases, and it is the focus of our knowledge-acquisition research.

3.1 KI: A Tool for Knowledge Integration

Knowledge integration is the process of incorporating new information into an existing knowledge base; it involves determining how the new information interacts with the existing knowledge. For the past three years we have been constructing KI, a tool that performs knowledge integration as it helps a domain expert extend the Botany Knowledge Base.

When provided with new information, KI retrieves relevant knowledge from the knowledge base and uses it to critique the new information. This involves identifying the ways in which existing knowledge corroborates or conflicts with new information. An example of KI integrating new information into the Botany Knowledge Base is described in Figure 1.[1]

KI goes beyond identifying "surface" inconsistencies, such as explicit constraint violations, by determining subtle interactions between new information and existing knowledge. This requires a focused, best-first search exploring the consequences of new information. KI's model of knowledge integration comprises three prominent activities:

1. Recognition: identifying the knowledge relevant to the new information.

[1]KI does not generate and parse natural language; this example has been converted from a language of frames, slots, and values.

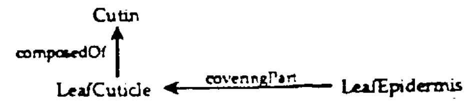


Figure 2: New information describing Leaf Cuticle

2. Elaboration: applying the expectations provided by relevant knowledge to determine the consequences of the new information.
3. Adaptation: modifying the knowledge base to accommodate the elaborated information.

3.1.1 Recognition

During recognition KI identifies concepts in the knowledge base that are relevant to the new information. This involves maintaining a *learning context* – a set of propositions about concepts deemed relevant to the new information. When presented with new information, KI initializes the context with the new information; Figure 2 shows the context initialized with the information from the first line of Figure 1. To extend the learning context, KI uses viewpoints to determine which concepts in the knowledge base, beyond those explicitly referenced in the context, are relevant.

Viewpoints are sets of propositions that interact in some significant way and should therefore be considered together. Viewpoints are created by applying a generic *view type* to a domain concept. Each view type is a parameterized semantic net, represented as a set of paths emanating from a root node. Applying a view type to a concept involves binding the concept to the root node and instantiating each path. Figures 3a and b present an example view type and the viewpoint created by applying it to leaf epidermis.

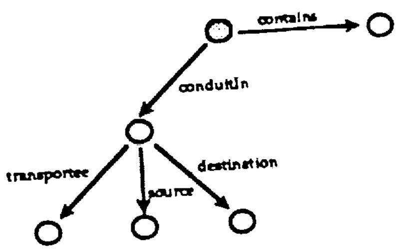
To extend the learning context, KI finds the viewpoints that contain concepts already in the learning context. Each candidate viewpoint is scored with a heuristic measure of relevance: the percentage of concepts contained in the viewpoint that are also contained in the learning context. KI presents the list of candidate viewpoints, ordered by their relevance score, to the domain expert, who selects one for use.[2] The set of propositions contained in the selected viewpoint are added to the learning context. This results in a learning context containing those concepts in the knowledge base considered most relevant to the new information.

3.1.2 Elaboration

During elaboration KI determines how the new information interacts with the existing knowledge within the learning context. Rules in the knowledge base are allowed to exhaustively

[2] Alternatively, an autonomous version of KI selects the viewpoint having the highest relevance score.

3a) Qua Container



3b) Leaf Epidermis Qua Container

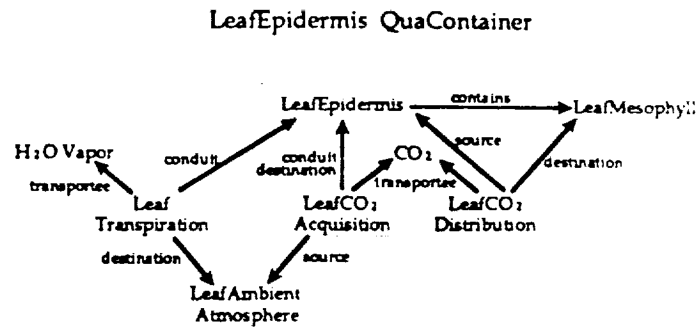


Figure 3: *(a):* The view type *Qua Container* identifies properties that are relevant to an object's function as a container. These properties include the contents of the container and the processes that transport items into and out of the container. *(b):* Applying this view type to Leaf Epidermis identifies the segment of the knowledge base that represents a Leaf Epidermis in its role as a container. For example, this segment includes propositions representing that Leaf Transpiration is a process by which water vapor is transported from inside the Leaf Epidermis to the atmosphere outside of the Leaf Epidermis.

forward-chain, propagating the consequences of the new knowledge throughout the context. For example, one consequence of a leaf having a leaf cuticle is that the leaf epidermis is impermeable to gases. Some of the domain inference rules applicable to this example are listed in Figure 4, and the resulting conclusions are presented in Figure 5.

KI enters a cycle of recognition (*i.e.*, selecting viewpoints) and elaboration (*i.e.*, applying inference rules) that explicates the consequences of the new information. The propositions added to the learning context during recognition determine which implicit consequences of the new information will be made explicit during elaboration. This cycle continues until the user intervenes or the relevance scores of all candidate viewpoints fall below a threshold. Figures 6 and 7 illustrate the second round of this cycle. The recognition phase extends the context of Figure 5 with the set of propositions describing how the leaf acquires and makes use of carbon dioxide. The elaboration phase propagates the consequences of the new information throughout the extended context.

3.1.3 Adaptation

During adaptation, KI appraises the inferences completed during elaboration and assists the user in modifying the knowledge base to accommodate the consequences of the new information. This can involve extending or retracting existing knowledge structures, or it can involve eliciting additional knowledge from the domain expert.

rule 1: If an object is composed of cutin,
then it is impermeable to gases.

rule 2: If the covering part of an object is impermeable to a substance,
then the object is impermeable to the substance.

rule 3: If the conduit is impermeable to the transportee,
then the transportation event is disabled

rule 4: If resource acquisition is disabled,
then resource distribution is also disabled.

rule 5: If either resource acquisition or distribution are disabled,
then resource provision is also disabled.

rule 6: If resource provision is disabled,
then resource utilization is also disabled.

rule 7: If either resource provision or utilization are disabled,
then resource assimilation is disabled.

rule 8: If leaf photosynthesis is disabled,
then the leaf is starving.

Figure 4: Example inference rules

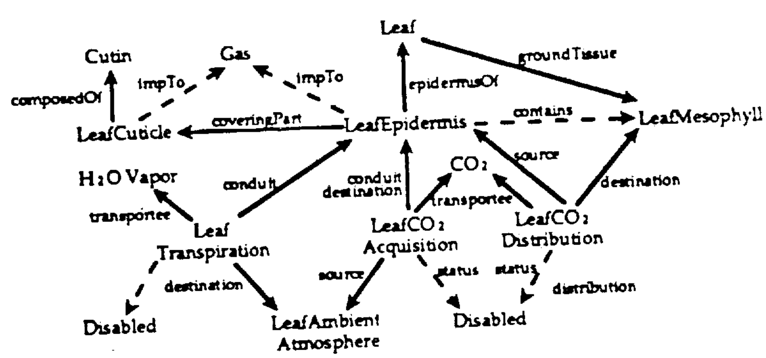


Figure 5: Rules in the knowledge base are used to propagate the consequences of the new information throughout the context of Figure 3b. The dashed lines indicate propositions that are computed during elaboration. For example, since the epidermis is impermeable to gases, carbon dioxide cannot be transported through the epidermis; therefore, the leaf cannot acquire carbon dioxide (see rule 3 of Figure 4).

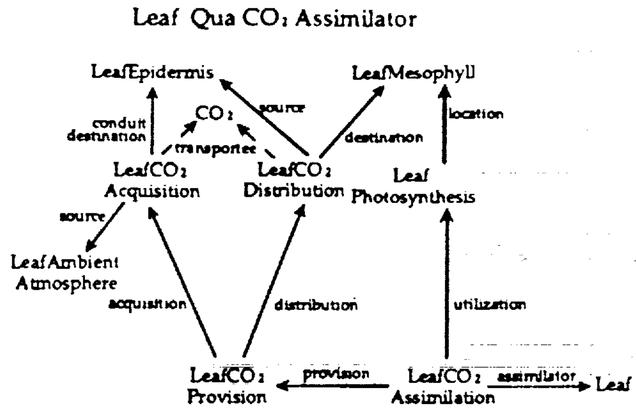


Figure 6: *Carbon Dioxide Qua Leaf Assimilate* This segment of the knowledge base represents the process by which a leaf acquires and uses carbon dioxide. For example, the leaf acquires carbon dioxide from the atmosphere and uses it during photosynthesis. The learning context of Figure 5 is extended with these propositions during the second round of recognition using the viewpoint "Leaf *qua* CO_2 assimilator."

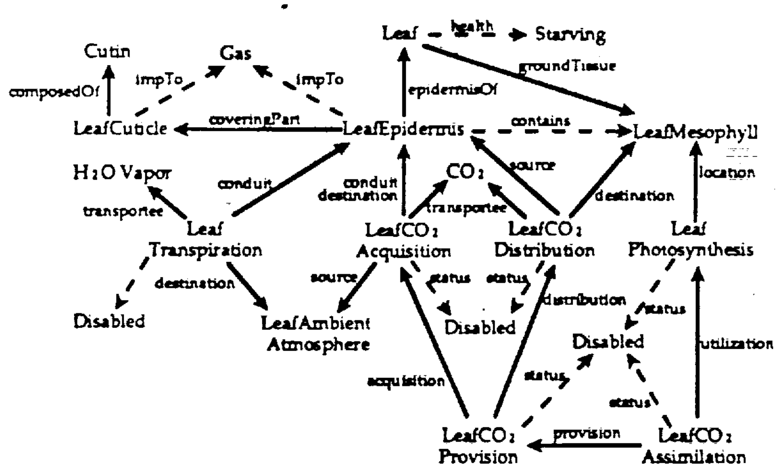


Figure 7: During the second round of elaboration, rules in the knowledge base are used to propagate the consequences of the new information throughout the extended learning context. For example, since the leaf cannot acquire carbon dioxide, photosynthesis cannot occur (see rules 5 and 6 of Figure 4).

In the example, elaboration reveals that the leaf cuticle benefits the leaf by restricting water loss through transpiration. The explanation supporting this conclusion can be generalized to suggest that other organs of a plant's shoot system (*e.g.*, stems, fruit) will also benefit from having a cuticle, and KI suggests this generalization to the domain expert.

Elaboration also reveals that the leaf's cuticle prevents the leaf from acquiring carbon dioxide from the atmosphere. Since carbon dioxide is an essential resource for photosynthesis, KI concludes that leaves having cuticle cannot perform photosynthesis. This conflicts with the expectation that leaves, in general, must be able to perform photosynthesis. To resolve this conflict, KI identifies plausible modifications to the knowledge base that would allow the leaf to acquire carbon dioxide and perform photosynthesis. These suggestions prompt the domain expert to provide additional information describing *stomata*, portals in the leaf's epidermis that allow restricted gas flow between the atmosphere and the leaf's interior.

This example illustrates how a tool for knowledge integration helps a domain expert develop a knowledge base. The tool identifies gaps and inconsistencies in the knowledge base and adapts it to accomodate new information. Automating these activities is critical for developing large, multifunctional knowledge bases because changes to the knowledge base can have significant, unforeseen consequences.

4 Access and Use of Multifunctional Knowledge Bases

We have developed prototype software for answering questions using a multifunctional knowledge base. Given a knowledge base and a student's question, an answer is generated in two steps:

- *content determination*: select or infer the portion of domain knowledge constituting a correct and coherent response.
- *text generation*: arrange the information into a linear sequence of propositions and express the propositions in natural language.

The following sections discuss the types of questions to be answered, methods for answering questions, and the results of applying our prototype question-answering system to the Botany Knowledge Base [29].

4.1 Question Types

A *question type* is a template for a class of questions that have similar conceptual representations and that can be answered using the same methods. For example, the question "What is a chloroplast?" belongs to the *definition* question type, and the question "How does a petal differ from a sepal?" belongs to the *comparison* question type. Question types are important for intelligent tutoring because they capture the range of questions that a

Question Type	*Meaning*	*Examples*
Definition	Describe important aspects	What is a chloroplast?
Comparison	Describe similarities or differences	How does a petal differ from a sepal?
Why	Describe causes or resulting states	Why are plants green? Why do plants absorb CO_2?
Why not	Describe preventions or missing causes	Why don't fungi contain chloroplasts?
Hypothetical	Describe important results of given conditions	What if a seed had no endosperm?

Table 1: A Small Sample of the Question Types

student can ask and they organize the automated reasoning strategies needed to answer the questions.

Our set of question types is similar to the thirteen conceptual categories of questions proposed by Lehnert [13], and subsequently extended by Hughes [9]. However, we have added question types concerning the physical structure of objects, the roles of objects in processes, and hypothetical situations. Table 1 is a small sample of our question types, and [2] provides a complete description.

4.2 Content Determination

The first step in answering a question is *content determination*: selecting the information that should be contained in a response. There is considerably more information in a knowledge base of fundamental knowledge than should be presented in a coherent response.

A common approach to the problem of selecting knowlege is to use *viewpoints*, which are collections of facts that belong together [37, 20, 18, 35]. For example, the viewpoint of "photosynthesis as production" contains facts about the producer, the products, and the raw materials, of photosynthesis. By contrast, the viewpoint of "photosynthesis as energy transduction" describes the input and output energy forms.

Most researchers have assumed that viewpoints are explicitly encoded in the knowledge base. For example, viewpoints in Swartout's Xplain system consist of annotations on elements of domain knowledge. The annotations indicate when a piece of knowledge should be included in an explanation. Similarly, viewpoints in McKeown's Advisor system (called perspectives) are represented by multiple hierarchies, each representing a single perspective. Finally, viewpoints in McCoy's system (also called perspectives) are represented by lists associated with each object in the knowledge base. Each list specifies the salience of each of the object's properties under a particular perspective.

Despite the emphasis on this approach, explicitly representing viewpoints for a large-scale knowledge base is infeasible. For example, Figure 1 illustrates the viewpoints of "photosynthesis as production" and "photosynthesis as energy transduction." In addition to these viewpoints, some circumstances require viewing photosynthesis as CO_2 utilization, a process requiring chlorophyll, and a biosynthesis enabling process. From just the information in Figure 1, dozens of viewpoints are possible.

Our solution to this problem is to dynamically generate viewpoints when they are needed to answer particular questions. This is done using a small number of ***view types*** that determine patterns of knowledge structures constituting viewpoints. First we describe the view types, then we explain how view types are used to generate viewpoints. A more comprehensive description of these issues is in [2].

4.2.1 View Types

We believe that a small number of view types — such as categorical, structural, functional, and modulatory — are sufficient to characterize all viewpoints within the physical sciences. Our support for this conjecture is preliminary but encouraging. First, we found these view types and their combinations sufficient to generate adequate definitions for over fifty terms chosen at random from the glossary of a botany textbook. Second, as described below, we have successfully used view types in our prototype question-answering system. We will continue investigating the adequacy of these view types for answering a wide range of questions, and we will extend them as required.

The *categorical* view type emphasizes the properties and relationships that indicate how a concept is a special case of one of its generalizations in a class hierarchy. For example, "flower as reproductive organ" is a categorical viewpoint. This viewpoint includes the particular reproductive parts of the flower (because reproductive organs have reproductive parts) as well as the reproductive processes in which it participates (because reproductive organs participate in reproductive processes).

The *structural* view type emphasizes an object's subparts (*substructural* view type) and superparts (*superstructural* view type). A substructural viewpoint of a seed contains the knowledge that a seed consists of the endosperm and the embryo, both of which are contained by the seed coat. A superstructural viewpoint of an endosperm contains the knowledge that the endosperm is a part of the seed contained in the seed coat. As illustrated by these examples, a structural viewpoint includes those relationships that specify how the parts are interconnected.

In addition to describing the *physical* structure of objects, the structural view type also describes the *temporal* structure of entities and processes. The temporal substructure of an entity is the stages it goes through during its existence. The substructure of a process is its steps, or *subevents*. For example, a temporal substructural viewpoint is "photosynthesis consists of the light reactions followed by the dark reactions." Temporal superstructural viewpoints also belong to the structural view type.

The *functional* view type emphasizes the role of an object in a process. By definition, it includes some kind of *actor* relationship, such as *producer, agent*, or *raw material*. For example, the viewpoint "chloroplast as the producer in plant photosynthesis" belongs to the functional view type. Although this example illustrates a direct relationship between an object and a process, sometimes the relationship is indirect. A part or specialization of the object may be the actor, rather than the object itself. For instance, one function of a seed is to protect the plant embryo, although strictly speaking it is the seed coat, a part of the seed, that protects the embryo.

The *modulatory* view type emphasizes how one object or process affects (or is affected by) another object or process. A modulatory viewpoint necessarily includes modulatory relationships, such as *causes, prevents, enables,* or *facilitates*. Other information also may be included, as with the functional view type. Examples of modulatory viewpoints are "sunlight as a requirement for plant growth" and "embryo growth as a cause of seed coat rupture."

4.2.2 Using View Types to Answer Questions

A question-answering system uses view types for content determination by first using them to select viewpoints from the knowledge base and then using the selected viewpoints to construct a response.

To isolate a particular viewpoint from the knowledge base, a question-answering system first selects the *concept of interest* which is the main topic of the viewpoint and is determined by the student's question. The system then selects an appropriate view type for the question at hand. This is done using heuristic rules that specify, for each question type, which view types are most useful for generating answers to questions of that type. These heuristics are sensitive to the kinds and amount of knowledge associated with the concept of interest in the knowledge base.

After the view type has been selected, the system selects the *reference concept* to which the concept of interest should be related. It serves as an anchor point for relating new information to what the student already knows.

A view type, when applied to a concept of interest and a reference concept, specifies the viewpoint to be selected from the knowledge base. For example,

- View Type: Functional
- Concept of Interest: Pollen
- Reference Concept: Plant Reproduction

specifies the viewpoint "the functional role of pollen in plant reproduction."

Once the system has determined the concept of interest, view type, and reference concept, it uses a content determination strategy to select the specified viewpoint from the knowledge base. After selecting the viewpoint from the knowledge base, the system uses the viewpoint (possibly together with other viewpoints) as the basis of a response. The way in which the viewpoint is used depends upon the type of question. For definition questions, the selected viewpoint(s) can be used directly as the content of a response. For comparison questions, the similarities and differences in the selected viewpoints constitute the content of the response.

4.3 Text Generation

After selecting the content of a response, a question answering system must express it in English. This process of translating from the internal representation of the knowledge base into grammatical text is called text generation. Fortunately, domain-independent computer programs for text generation are available, and we plan to integrate one of these programs with our tutoring software.

Two major projects on text generation have produced useful systems. The Mumble system [19] generates text from specifications provided by a content-determination module or text planner. A text specification is a conceptual (non-linguistic) description of what should be said, how it should be structured, and what perspective or emphasis it should reflect. A specification is expressed in terms of the internal conceptual representation of the underlying knowledge base. To generate text from specifications, Mumble uses knowledge of how objects in the knowledge base correspond to possible syntactic structures and phrases. Each element of a text specification is associated with a set of such choices and a decision procedure for selecting among them. Mumble is fast and portable, and has been successfully used as the realization component for several systems, including Romper [11] and Text [20].

Another portable text generator is Penman [16]. Like Mumble, Penman makes a clear distinction between the domain-dependent and domain-independent system modules. Penman produces text from a hierarchical text plan that specifies content and organization. Using one of the largest English grammars encoded on a computer, Penman can be used for a variety of domains and knowledge representations. Penman's designers claim that its techniques are adequates for use with several existing explanation generation systems, including Text [20], Proteus [7], and KDS [17].

4.4 Results of Our Prototype Question-Answering System

We have built a prototype system, called Prosaiq, that answers questions using the Botany Knowledge Base. Currently, the system answers questions that are classified as definition and comparison question types using the categorical, structural, and functional view types. The following examples demonstrate the use of view types to select information comprising a coherent response to the definition question "What is photosynthesis?"

When the chosen view type is categorical and the chosen reference concept is *Biological Production*, Prosaiq generates[3]

> Photosynthesis is a biological production event in which a photosynthetic organ converts the raw materials carbon dioxide and water into the product glucose

[3]The system's output has been manually translated into English for these examples.

and the byproduct oxygen. It consists of the light reactions followed by the dark reactions.

To generate this definition, Prosaiq selects only those relations of *Photosynthesis* that are inherited from *Biological Production* or one of its generalizations. Although this is a small portion of the knowledge associated with *Photosynthesis*, it is a coherent definition because it adheres to a particular viewpoint (photosynthesis as production).

The next example illustrates using the categorical view type to answer the comparison question:

How are photophosphorylation and cellular respiration alike?

When the chosen reference concept is *Biological Production*, Prosaiq generates:

Photophosphorylation and cellular respiration are alike in that they are both biological production events in which the end product is ATP.

Photophosphorylation and respiration have many similarities; many of these arise because both processes are a kind of biological production. By using the categorical view type and making the assumption that the student knows about biological production, the system generates a concise response containing only the similarities that are most likely to be new to the student.

4.5 Discourse Planning

Building on the ability to answer questions, we are developing a prototype system for planning and generating extensive pedagogical discourses. Just as coherence is an issue in answering questions, it is also important for planning a discourse. A discourse planner must ensure that both the knowledge that is selected and the manner in which it is organized is coherent for the student. In contrast to a question answerer, a planner must address three additional issues. First, it must maintain coherence across much longer passages of text. Second, it should take advantage of opportunities to educate the student about important concepts in the domain, and must weave these discussions into the discourse in a coherent manner. Third, it must allow the student to interrupt to ask questions, and then replan the remainder of the discourse as needed to maintain coherence.

The discourse planning task is formulated as follows:

- Given:
  - a discourse goal
  - domain knowledge
  - the student's current state of knowledge
- Generate:

- a discourse that achieves the goal, includes the domain knowledge appropriate for the student, and is organized in a manner that is appropriate for the student
- an updated student model that reflects what the student has been told

The discourse goal can be furnished by either the student or by an instructional planner, such as those proposed by Woolf and McDonald [39], Peachy and McCalla [28], and Murray [26]. The domain knowledge is contained in the knowledge base. The student's current state of knowledge is maintained in the *student model.*

In addition to the issues faced by a question answerer, an effective discourse planner must address three additional issues: global coherence, opportunistic pedagogy, and interruptability.

A discourse planner must maintain *global coherence* across much longer passages of text than a question-answerer. There are several aspects of global coherence that should be incorporated in a discourse planner. First, a discourse planner should cluster semantically similar knowledge together and order these clusters by their prerequisites. Second, it should provide organizational aids such as an outline early in the discourse and a summary at the end of the discourse. Finally, a discourse planner should maintain thematic coherence across a discourse. For example, when planning a discourse on photosynthesis, a planner should adhere to a theme of either photosynthesis viewed as production or photosynthesis viewed as energy transduction throughout the discourse.

In addition to maintaining global coherence, an effective discourse planner must address the issue of *opportunistic pedagogy.* As it plans a discourse, it should take advantage of opportunities to educate the student about concepts in the domain that are closely related to the topic but are unknown to the student. In general, a planner should not only notice these opportunities and take advantage of them, it should actively seek them, while avoiding unnecessary digressions.

Finally, a discourse planner should be *interruptable.* An important goal of intelligent tutoring systems research for twenty years has been to provide *mixed-initiative* instruction [3]. In a mixed-initiative environment, both the student and the system may direct the tutorial exchange. To provide such an environment, the planner must allow the student to interrupt the discussion to ask a question. Interruptability presents a significant problem for a discourse planner. By responding to the student's question in the middle of a discourse, the planner may need to radically change how it should complete the discourse. For example, its response to the question may obviate the need for introducing concepts that are to appear later in the discourse. In short, providing interruptability implies that the planner must dynamically revise its plans.

We are designing a discourse planner that addresses the issues of global coherence, opportunistic pedagogy, and interruptability by using a *delayed-commitment* approach to plan construction. This approach increases the flexibility of a planner by decoupling content determination from organization.

To generate a discourse plan, our planner adds elements to a loosely organized workspace, and gradually imposes structure on them. When the plan elements are totally ordered, they are passed to the text generator for conversion to text.

By decoupling content determination from organization, the order in which the planner constructs the elements is different than the order in which the utterances derived from those elements appear in the discourse. This decoupling permits more flexibility than current approaches to planning which use discourse strategies. At each step of the strategy, these planners extract a fragment of the knowledge base and translate it into text [20]. Although strategies in some planners can invoke other strategies [27], the global organization of the discourse is largely determined by the order of the steps in the strategies.

The delayed-commitment approach to discourse planning promotes global coherence. As the planner constructs the plan elements, it can organize them according to their estimated familiarity to the student. In contrast, with current planning systems, the designer of the system must anticipate in advance what concepts will be familiar to the student, and embed these decisions in its strategies. For example, suppose the system were planning a discourse on the process of embryo sac formation. If the student were familiar with the concept of double fertilization, a process following embryo sac formation, then the planner could explain this conceptual link to a familiar concept early in the discourse. On the other hand, if the student were unfamiliar with double fertilization, the planner could either omit this discussion or postpone it until later in the discourse.

The delayed-commitment approach promotes opportunistic pedagogy by allowing the planner to interject discussions of unexplained, but important, concepts and to restructure the discourse as needed. For example, suppose the planner was explaining embryo sac formation and its two primary actors: a megaspore, which is haploid, and a megaspore mother cell, which is diploid. Because these cell types are important, the planner should digress and explain their differences. Moreover, rather than interjecting this discussion in the middle of another topic, the planner can relocate it to an appropriate place in the discourse. In contrast, current planners cannot effectively take advantage of pedagogical opportunities because they cannot reorganize the discussion. For them, the global organization is fixed in advance.

The delayed-commitment approach also promotes interruptability by permitting plan revision. After responding to a question, the planner can reorganize the remainder of the discourse. For example, suppose the system were discussing reproduction in angiosperms and the student asked about the related concept of "alternation of generations." Then after answering the question, the planner could replan the remainder of the discourse to relate the upcoming concepts to the alternation of generations. In contrast, current planners cannot dynamically revise their plans. The ability to reorder plan elements rather than being forced to follow a pre-defined strategy permits a much higher degree of flexibility than is allowed by current planners.

5 Summary

Multifunctional knowledge bases offer a significant advance in artificial intelligence because they can support numerous expert tasks within a domain. As a result they amortize the costs of building a knowledge base over multiple expert systems and they reduce the brittleness of each system.

Due to the inevitable size and complexity of multifunctional knowledge bases, their construction and maintenance require knowledge engineering and acquisition tools that can automatically identify interactions between new and existing knowledge. Furthermore, their use requires software for accessing those portions of the knowledge base that coherently answer questions.

We have made considerable progress in developing software for building and accessing multifunctional knowledge bases. We have developed a language for representing knowledge, software tools for editing and displaying knowledge, a machine learning program for integrating new information into existing knowledge, and a question-answering system for accessing the knowledge base.

In our continuing research, we plan to significantly improve these prototype systems and to integrate them into a single framework. The resulting software environment will be effective for building, maintaining, and using large multifunctional knowledge bases in any domain.

References

[1] AAAI workshop on very large multifunctional knowledge bases, 1988. Held at the National Conference on Artificial Intelligence, St. Paul, MN, August 22, 1988. Organizing Committee: Carbonell, J. Rich, E., Thomason, R., Nirenburg, S. and Monarch, I.

[2] L. Acker, J. Lester, A. Souther, and B. Porter. Generating coherent explanations to answer students' questions. In H. Burns and J. Parlett, editors, *Intelligent tutoring systems: Evolutions in design*. Lawrence Erlbaum, 1990. Hillsdale, NJ.

[3] J.R. Carbonell. AI in CAI: An artificial intelligence approach to computer-assisted instruction. *IEEE Transactions on Man-Machine Systems*, 11(4):190–202, 1970.

[4] Carnegie Group. *KnowledgeCraft CRL Technical Manual, Version 3.1*, 1987.

[5] B. Chandrasekaran and S. Mittal. Deep versus compiled knowledge approaches to diagnostic problem solving. *International Journal of Man-Machine Studies*, pages 425–436, 1983.

[6] W. Clancey and R. Letsinger. Neomycin: Reconfiguring a rule-based expert system for application to teaching. In *Seventh International Joint Conference on Artificial Intelligence*, pages 829–836, 1981.

[7] A. Davey. *Discourse Production*. Edinburgh: Edinburgh University Press, 1979.

[8] P. Fink, J. Lusth, and J. Duran. A general expert system design for diagnostic problem solving. *IEEE 1984 Proceedings of the Workshop on Principles of Knowledge-Based Systems*, 1984.

[9] S. Hughes. Question classification in rule-based systems. *Expert Systems*, pages 123–131, 1986.

[10] IntelliCorp. *KEE Software Development System User's Manual, Version 3.0*, 1985.

[11] R. Karlin. Romper mumbles. Technical Report MS-CIS-85-41, Department of Computer Science, University of Pennslyvania, 1985.

[12] J. Larkin, F. Reif, J. Carbonell, and A. Gugliotta. Fermi: A flexible expert reasoner with multi-domain inferencing. *Cognitive Science*, 12:101–138, 1988.

[13] W. Lehnert. A conceptual theory of question answering. In B. Grosz, K. Jones, and B. Webber, editors, *Reading in Natural Language Processing*. Morgan Kaufmann, 1986.

[14] D. Lenat and R. Guha. *Building Large Knowledge Based Systems*. Reading, MA:Addison-Wesley, 1990.

[15] D. Lenat, M. Prakash, and M. Shepherd. Cyc: Using common sense knowledge to overcome brittleness and knowledge acquisition bottlenecks. *AI Magazine*, 6(4):65–85, 1983.

[16] W. Mann. An overview of the Penman text generation system. In *Proceedings of AAAI*, pages 261–265, 1983.

[17] W. Mann and J. Moore. Computer generation of multiparagraph english text. *American Journal of Computational Linguistics*, 7(1):17–29, 1981.

[18] K. McCoy. The role of perspective in responding to property misconceptions. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 791–793, 1985.

[19] D. McDonald. Description directed control: Its implications for natural language generation. *Computers and Mathematics*, 9(1):111–130, 1983.

[20] K. McKeown. Discourse strategies for generating natural language text. *Artificial Intelligence*, 27:1–42, 1985.

[21] K. McKeown. Tailoring explanations for the user. Technical Report CUCS-172-85, Columbia University, 1985.

[22] W. Mettrey. An assessment of tools for building large knowledge-based systems. *AI Magazine*, 8(4):81–89, 1987.

[23] T. Mitchell, J. Allen, P. Chalasani, J. Cheng, O. Etzioni, M. Ringuette, and J. Schlimmer. Theo: A framework for self-improving systems. In K. VanLehn, editor, *Architectures for Intelligence*. Lawrence Erlbaum Associates, 1989.

[24] J. Mostow and W. Swartout. Towards explicit integration of knowledge in expert systems: An analysis of Mycin's therapy selection algorithm. In *Proceedings of the National Conference on Artificial Intelligence*, pages 928–935, 1986.

[25] K. Murray and B. Porter. Controlling search for the consequences of new information during knowledge integration. In *Proceedings of the Sixth International Workshop on Machine Learning*, 1989.

[26] William R. Murray. Control for intelligent tutoring systems: A blackboard-based dynamic instructional planner. *AICOM*, 2(2), 1989.

[27] C. Paris. Tailoring object descriptions to a user's level of expertise. *Computational Linguistics*, 14(3):64–78, 1988.

[28] D.R. Peachy and G.I. McCalla. Using planning techniques in intelligent tutoring systems. *International Journal of Man-Machine Studies*, 24:77–98, 1986.

[29] B. Porter, J. Lester, K. Murray, K. Pittman, A. Souther, L. Acker, and T. Jones. AI research in the context of a multifunctional knowledge base. Technical Report AI88-88, University of Texas at Austin, 1988.

[30] M. Richer. An evaluation of expert system development tools. *Expert Systems*, 3(3):166–183, 1986.

[31] E. Schoen and R. Smith. Impulse: A display-oriented editor for Strobe. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, pages 855–858, 1983.

[32] R. Smith. Strobe: Support for structured object knowledge representation. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, 1983.

[33] E. Soloway, J. Bachant, and K. Jensen. Assessing the maintainability of Xcon-in-Rime: Coping with the problems of a very large rule-base. *Proceedings of the Sixth National Conference on Artificial Intelligence, Seattle, Washington*, 1987.

[34] A. Souther, L. Acker, J. Lester, and B. Porter. Using view types to generate explanations in intelligent tutoring systems. In *Proceedings of the 11th Cognitive Science Society Conference*, 1989.

[35] D. Suthers. Perspectives in explanation. Technical Report COINS-89-24, Department of Computer Science, University of Massachusetts at Amherst, 1988.

[36] W. Swartout. Xplain: A system for creating and explaining expert consulting programs. *Artificial Intelligence*, 21:285–325, 1983.

[37] W. Swartout. Xplain: A system for creating and explaining expert consulting programs. *Artificial Intelligence*, 21(3):285–325, 1983.

[38] P. Szolovits. Expert system tools and techniques: Past, present and future. In W. Grimson and R.S. Patil, editors, *AI in the 1980's and Beyond: An MIT Survey*, pages 43–74. Cambridge, MA: MIT Press, 1987.

[39] B. Woolf and D. McDonald. Context-dependent transitions in tutoring discourse. In *Proceedings of the National Conference on Artificial Intelligence*, pages 355–361, 1984.