[44] Peter F. Patel-Schneider. Undecidability of subsumption in NIKL. *Artificial Intelligence*, 39(2):263–272, 1989.

[45] 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: The Botany Knowledge Base project. Technical Report AI88-88, University of Texas at Austin, 1988.

[46] P. Raven, R. Evert, and H. Curtis. *Biology of Plants.* Worth Publishers, New York, 1976.

[47] Jeff Rickel and Bruce Porter. Automated modeling for answering prediction questions: Exploiting interaction paths. In *Proceedings of the Sixth International Workshop on Qualitative Reasoning, Heriot-Watt University, Edinburgh, Scotland*, 1992.

[48] Manfred Schmidt-Schaub. Subsumption in KL-ONE is undecidable. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 421–431. Morgan Kaufmann, May 1989.

[49] Penelope Sibun. *Locally Organized Text Generation.* PhD thesis, Department of Computer Science, University of Massachusetts at Amherst, 1992.

[50] A. Souther, L. Acker, J. Lester, and B. Porter. Using view types to generate explanations in intelligent tutoring systems. In *Proceedings of the Eleventh Cognitive Science Conference*, pages 123–130, 1989.

[51] John F. Sowa. *Conceptual Structures.* Addison-Wesley, Reading, MA, 1984.

[52] Albert Stevens and Cindy Steinberg. A typology of explanations and its applications to intelligent CAI. Technical Report TR 4626, Bolt Beranek and Newman, Inc., 1981.

[53] Daniel D. Suthers. Providing multiple views of reasoning for explanation. In *Proceedings of the International Conference on Intelligent Tutoring Systems*, pages 435–442, 1988.

[54] Daniel D. Suthers. Perspectives in explanation. Technical Report COINS-89-24, Department of Computer Science, University of Massachusetts at Amherst, 1989.

[55] Daniel D. Suthers. The epistemological structure of explanations. In *Proceedings of the AAAI-90 Workshop on Explanation*, pages 178–187, July 1990.

[56] Daniel D. Suthers. Task-appropriate hybrid architectures for explanation. In *Proceedings of the AAAI-91 Workshop on Comparative Analysis of Explanation Planning Architectures*, pages 80–94, July 1991.

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

[58] William A. Woods. Understanding subsumption and taxonomy: A framework for progress. In John F. Sowa, editor, *Principles of semantic networks: Explorations in the representation of knowledge*, pages 45–94. Morgan Kaufmann, San Mateo, CA, 1991.

[59] William A. Woods and James G. Schmolze. The KL-ONE family. Technical Report TR-20-90, Harvard University Center for Reserach in Computing Technology, 1990.

[29] William C. Mann and Sandra A. Thompson. Rhetorical structure theory: A theory of text organizations. Technical Report ISI/RS-87-190, Information Sciences Institute, University of Southern California, 1987.

[30] Suzanne McBride. A graphic access and presentation tool for related knowledge base concepts. Master's thesis, Department of Computer Sciences, University of Texas at Austin, 1992.

[31] Kathleen 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.

[32] Kathleen McCoy. Generating context-sensitive responses to object-related misconceptions. *Artificial Intelligence*, 41:157–195, 1989.

[33] Kathleen R. McKeown. *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text.* Cambridge University Press, New York, 1985.

[34] Kathleen R. McKeown. Generating goal-oriented explanations. *International Journal of Expert Systems*, 1(4):377–395, 1988.

[35] Johanna D. Moore and William R. Swartout. A reactive approach to explanation. In *Proceedings of the Fourth International Workshop on Natural Language Generation*, pages 17–21, July 1988.

[36] Johanna D. Moore and William R. Swartout. A reactive approach to explanation. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1504–1510, 1989.

[37] Kenneth S. Murray and Bruce W. Porter. Controlling search for the consequences of new information during knowledge integration. In *Proceedings of the Sixth International Workshop on Machine Learning*, pages 290–295, San Mateo, CA, 1989. Morgan Kaufmann.

[38] Bernhard Nebel. Computational complexity of terminological reasoning in BACK. *Artificial Intelligence*, 34(3):371–383, 1988.

[39] Gordon S. Novak Jr. Representations of knowledge in a program for solving physics problems. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, pages 286–291, August 1977.

[40] Gordon S. Novak Jr. and Agustin A. Araya. Physics problem solving using multiple views. Technical Report TR-173, The University of Texas at Austin, March 1981.

[41] Cecile L. Paris. Description strategies for naive and expert users. In *Proceedings of the Twenty-third Annual meeting of the Association for Computational Linguistics*, pages 238–245, 1985.

[42] Cecile L. Paris and Kathleen R. McKeown. Discourse strategies for describing complex physical objects. In Gerard Kempen, editor, *Natural Language Generation*, pages 97–115. Martinus Nijhoff, Dordrecht, The Netherlands, 1987.

[43] Peter F. Patel-Schneider. A four-valued semantics for terminological logics. *Artificial Intelligence*, 38(3):319–351, 1989.

[12] Joseph E. Grimes. *The Thread of Discourse.* Mouton, The Hague, Paris, 1975.

[13] Barbara J. Grosz. The representation and use of focus in a system for understanding dialogs. In B. Grosz, K. Jones, and B. Webber, editors, *Readings in Natural Language Processing*, pages 353–362. Morgan Kaufman, 1986.

[14] Barbara J. Grosz and Candace L. Sidner. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204, July-September 1986.

[15] R. V. Guha. Micro-theories and contexts in Cyc. Technical Report ACT-CYC-129-90, Microelectronics and Computer Technology Corporation, June 1990.

[16] R. V. Guha. Contexts: A formalization and some applications. Technical Report ACT-CYC-423-91, Microelectronics and Computer Technology Corporation, November 1991.

[17] Michael A. K. Halliday and Ruqaiya Hasan. *Cohesion in English.* Longman, London, 1976.

[18] Gary G. Hendrix. Expanding the utility of semantic networks through partitioning. In *Proceedings of the Fourth International Joint Conference on Artificial Intelligence*, pages 115–121, 1975.

[19] Jerry R. Hobbs. Towards an understanding of coherence in discourse. In Wendy Lehnert and Mark Ringle, editors, *Strategies for Natural Language Processing*, pages 223–243. Lawrence Erlbaum, Hillsdale, NJ, 1982.

[20] Jerry R. Hobbs. On the coherence and structure of discourse. Technical Report CSLI-85-37, Computer Science Department, Stanford University, 1985.

[21] Paul S. Jacobs. KING: A knowledge intensive natural language generator. In Gerard Kempen, editor, *Natural Language Generation*, pages 219–225. Martinus Nijhoff, Dordrecht, The Netherlands, 1987.

[22] Hyung J. Kook. A model-based representational framework for expert physics problem solving. Technical Report AI89-103, Artificial Intelligence Laboratory, The University of Texas at Austin, May 1989.

[23] B. Kuipers. Qualitative simulation. *Artificial Intelligence*, 29:289–338, 1986.

[24] George Lakoff and Mark Johnson. *Metaphors We Live By.* University of Chicago Press, 1980.

[25] Douglas B. Lenat and R. V. Guha. *Building Large Knowledge Based Systems.* Reading, MA:Addison-Wesley, 1990.

[26] James C. Lester and Bruce W. Porter. A revision-based model of instructional multi-paragraph discourse production. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*, pages 796–800, August 1991.

[27] James C. Lester and Bruce W. Porter. A student-sensitive discourse generator for intelligent tutoring systems. In *Proceedings of the International Conference on the Learning Sciences*, pages 298–304, 1991.

[28] Z. Liu and A. Farley. Shifting ontological perspectives in reasoning about physical systems. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, 1990.

The View Retriever will be evaluated more extensively when it supports our tutoring system for plant anatomy and physiology. It will be the primary method used by the tutor to access the Botany Knowledge Base as it constructs qualitative models [47] and generates explanations [26]. We are currently building this tutoring system, and we have found that knowledge base access at the level of viewpoints (as opposed to either individual facts or frames) greatly simplifies system design and implementation.

Acknowledgements

We would like to thank Paul Cohen for helping us evaluate the View Retriever. We would like to thank Jimi Crawford, Jeff Rickel, and James Lester for useful comments on earlier drafts of this paper. Finally, we would like to thank Art Souther for first realizing the importance of these access methods and helping to shape our approach.

References

[1] 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, Hillsdale, NJ, 1990.

[2] Liane E. Acker. *Access Methods for Large, Multifunctional Knowledge Bases.* PhD thesis, Department of Computer Sciences, University of Texas at Austin, 1992.

[3] Hippocrates G. Apostle. *Aristotle's Metaphysics.* The Peripatetic Press, Grinnell, Iowa, 1979.

[4] Paul Blair, R. V. Guha, and Wanda Pratt. Microtheories: An ontological engineer's guide. Technical Report CYC-050-92, Microelectronics and Computer Technology Corporation, March 1992.

[5] Daniel G. Bobrow and Terry Winograd. An overview of KRL, a knowledge representation language. *Cognitive Science*, 1:3–46, 1977.

[6] Ronald J. Brachman, Richard E. Fikes, and Hector J. Levesque. KRYPTON, a functional approach to knowledge representation. In Hector J. Levesque and Ronald J. Brachman, editors, *Readings in Knowledge Representation*, pages 412–429. Morgan Kaufman, Los Altos, CA, 1985.

[7] Ronald J. Brachman and James G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9:171–216, 1985.

[8] James Crawford. Access-limited logic—a language for knowledge representation. Technical Report AI90-141, University of Texas at Austin AI Laboratory, September 1990.

[9] Robert deBeaugrande. *Text Discourse and Process.* Ablex, Norwood, NJ, 1980.

[10] Brian Falkenhainer and Kenneth D. Forbus. Compositional modeling: Finding the right model for the job. *Artificial Intelligence*, 51:95–143, 1991.

[11] Kenneth D. Forbus. Qualitative process theory. *Artificial Intelligence*, 24:85–168, 1984.

The KI System Murray's KI is a tool for assisting knowledge enterers in making extensions to a large knowledge base [37]. Given a proposed knowledge-base extension, KI identifies how the new information violates or conforms to expectations arising from the existing knowledge. KI uses *views* to constrain the search for consequences of new information by applying inference methods to only the knowledge within the selected view(s). KI generates views using predefined *view types*.

This research extends KI's ability to dynamically generate viewpoints. Although KI's mechanism for generating views is general-purpose, the set of view types Murray provides is quite limited, as it is not intended to provide broad coverage. An emphasis of this work has been to provide a fairly complete set of viewpoint types useful in all physical domains.

3.7.3 Representing Viewpoints

Although the knowledge base from which a viewpoint is generated contains all of the facts the viewpoint comprises, representing the boundaries of the viewpoint itself in the knowledge base is also useful. This allows caching of commonly used viewpoints, and it provides a means for hand-coding viewpoints that are specific to a particular task or domain. Furthermore, storing a viewpoint in the knowledge base allows assertions about the viewpoint to be represented, such as the tasks for which it is likely to be useful or whether a user is expected to be familiar with its contents. Several suitable formalisms exist for representing viewpoints within a knowledge base, including Hendrix's *net spaces* [18] (as extended by Grosz [13]), Sowa's *perspectives* [51], Crawford's *views* [8], CycL's *multiple models* [25], and the most general approach, Guha's *contexts* [15, 4, 16].

4 Summary

The results of this research are methods for accessing information in large, structured knowledge bases. The advantages of our access methods compared with conventional methods are twofold.

First, our methods help users to locate concepts by providing an abstraction of the knowledge base, a content addressable, virtual knowledge base. A content addressable knowledge base allows users to locate a frame using a partial description of the frame's contents. A virtual knowledge base allows users to access concepts that are implicit in the knowledge base but are not reified as frames. Computing subsumption relationships is the crucial step in providing a content addressable, virtual knowledge base, and empirical evidence suggests that our approach is practical.

When an access method provides both content addressability and access to concepts in the virtual knowledge base, users need not know whether concepts are explicit in the knowledge base or where they are located. Users simply supply a description of the concept, embedded in an access request. If the concept has a frame associated with it, then the system will find and use that frame to service the request. Otherwise, the system will create and use a new frame. From the user's point of view, there is no distinction between accessing concepts by description and accessing concepts in the virtual knowledge base.

The second advantage of our access methods is that, after locating a concept, they extract viewpoints, coherent sets of facts that describe the concept from a particular perspective. We have identified many types of viewpoints and developed methods for extracting them from knowledge bases, either singly or in combinations. Our evaluation indicates that viewpoints extracted by our methods are comparable in coherence to those people construct.

system's attention on the knowledge that is most salient at a given point in the dialogue [13]. The KING system uses *views* to guide linguistic and conceptual choices that arise in natural language generation [21].

Other systems use knowledge similar to viewpoints to constrain automated reasoning. For example, Falkenhainer and Forbus use *perspectives* to construct models of physical devices [10]. Perspectives ensure that the model makes consistent simplifying assumptions and that the model is relevant to the reasoning task for which it is constructed. Perspectives also yield simplified models, which are more efficient to reason with, are more generally applicable (because they require less data), and yield more coherent explanations of problem-solving behavior.

The ability to take multiple viewpoints of an object is also important for solving physics problems. For example, the ISAAC and APEX problem solvers construct a formal representation of the given problem by viewing each object in the problem as a *canonical object*, an idealized or abstract object such as a point mass or a lever [39, 22]. Viewing actual objects as canonical objects is important because, while problems are stated in terms of complex, real-world objects, the principles and laws of physics are stated in terms of canonical objects. In addition, viewing a real-world object as a canonical object restricts the information about the actual object that may be used to solve the problem, which greatly reduces the size of the problem space [39, 40].

Finally, Algernon uses *views* for default reasoning [8]. Many default reasoning schemes are intractable, because they require the system to ensure that a default assumption is consistent with everything that is known before making that assumption. Restricting inference to the information found within a chosen view makes default reasoning more efficient.

3.7.2 Generating Viewpoints

Previous discussions of related work in sections 3.2, 3.3.2, and 3.5 described systems that generate viewpoints of one of the major types. This section discusses two systems, TEXT and KI, that are not limited to viewpoints of a single type.

The TEXT System McKeown's TEXT is a system that answers questions about the structure of a database [33]. TEXT treats the database schema as a source of domain knowledge and uses it to generate definitions, descriptions, and comparisons of domain concepts.

McKeown recognizes that to convey information about several properties of an entity, the system should not generate an arbitrary list of properties. To ensure coherence, the system should group together "properties that are in some way related to each other." Stated in the terminology used here, coherence requires organizing information according to viewpoints.

In an effort to structure knowledge according to viewpoints, TEXT uses the following heuristic (called a *focus constraint*) for selecting the next proposition to include in a response: choose the proposition with the greatest number of links to previously selected propositions. The result is a sequence of propositions in which each proposition is related to some preceding proposition. Although this heuristic increases the coherence of the responses TEXT generates, selecting propositions one at a time is not an optimal strategy for achieving viewpoint coherence. Because viewpoints often overlap, TEXT may unknowingly progress from one viewpoint to another related viewpoint without completing the first viewpoint. This results in fragmented or incomplete viewpoints, which degrade the coherence of the response. Using complete viewpoints, rather than atomic propositions, as the building blocks of a response yields greater coherence.

Source	*Coherence*	
	Mean	*Standard Deviation*
(1) Textbook Viewpoints	4.23	0.56
(2) View Retriever's Viewpoints	3.76	0.74
(3) Degraded Viewpoints	2.86	0.94
(4) Random Collections of Facts	2.62	0.86

Table 1: Ten judges rated the coherence of sets of facts from four sources (1=incoherent; 5=coherent). A statistical analysis using the T-test with 0.95 level of confidence shows no significant difference in coherence between sources (1) and (2) or between sources (3) and (4). There is a significant difference between all other pairs.

- the variance in coherence scores assigned by different judges, and
- the variance in coherence scores for passages from different sources (*e.g.,* textbooks, the View Retriever).

Thus, although judges varied in their harshness, they largely agreed on relative orderings.

In summary, we have not attempted to define "coherence." Instead, we have built access methods that purport to extract coherent viewpoints from large knowledge bases, applied them to our Botany Knowledge Base, and asked judges to compare the coherence of their viewpoints with collections of facts from other sources. This study determined that collections of facts vary significantly in their coherence and that viewpoints extracted by the View Retriever are comparable in coherence to textbook passages.

3.7 Related Work

This section discusses related work in three areas: applications of viewpoints, dynamically generating viewpoints, and representing viewpoints in a knowledge base.

3.7.1 Applications of Viewpoints

An important application of viewpoints is to ensure the coherence of the explanations that a question-answering or explanation-generation system produces. (Other sources of coherence, such as discourse coherence and textual coherence, also contribute to the overall coherence of an explanation.) In selecting the content of an explanation, it is important for a system to be sensitive to the current context, because the coherence of the explanation will be judged relative to that context. Different viewpoints provide different explanations of domain knowledge, each appropriate for different levels of expertise [54, 41, 42, 57], different user needs [34], different system goals [35, 36], and different dialogue contexts [31, 32, 35]. Suthers points out an additional benefit of constructing explanations from viewpoints rather than from atomic facts: accessing the knowledge base at the right level of abstraction (*i.e.,* at the viewpoint level) allows an explanation generator to concentrate on issues of discourse management, and it facilitates portability across domains and representational formalisms [53].

Viewpoints are also important for a variety of other applications, such as natural language processing. In discourse understanding, Grosz uses viewpoint-like knowledge structures called *focus spaces* to focus the

3.6 Evaluation of the View Retriever

The purpose of our evaluation was to measure the coherence of viewpoints the View Retriever extracts, as compared to the coherence of viewpoints found in human-generated text. For each of 12 topics in botany, sets of facts were drawn from 3 sources:

- a college-level botany textbook [46],
- the View Retriever applied to the Botany Knowledge Base, and
- facts selected randomly from a particular frame in the Botany Knowledge Base.

The viewpoints ranged in size from 3 to 11 facts. For each topic, textbook passages and random sets of facts were chosen to be roughly the same size as the viewpoint on that topic. Each group of facts (including the textbook passages) was translated manually into "simple English" to normalize presentation style.

Ten subjects (senior undergraduates and graduate students from the Botany and Biology Departments of the University of Texas at Austin) judged the coherence of several passages from each source. Subjects were given the following instructions:

> Each of the following pages contains a brief passage of text along with its subject. Please judge the coherence of the passage on a scale of 1 to 5. A passage should be scored "1" if it is seems no more coherent than a randomly selected group of facts on the subject. A passage should be scored "5" if it is as coherent as a passage of comparable length on the subject from a good textbook.
>
> Limit your consideration to the contents of each passage, and ignore issues of organization and rhetoric (such as writing style, wording, and diction). If you feel that the presentation of the material is poor, give the passage the same score that you would give a passage containing the same information but organized and presented in a better fashion.

Table 1 summarizes the subjects' responses. Statistical analysis (using a T-test with 0.95 level of confidence) yields the following results:

- The mean coherence of viewpoints from textbooks did not differ significantly from the mean coherence of viewpoints extracted by the View Retriever.
- The mean coherence of extracted viewpoints *did* differ significantly from the mean coherence of random collections of facts drawn from the same frame.

A further study gives additional evidence that the View Retriever extracts coherent viewpoints. Along with passages from the three sources described above, the subjects were given passages from a fourth source: viewpoints extracted by the View Retriever and then "degraded" by replacing some of their facts with randomly selected facts on the same topic. Twenty-eight such degraded viewpoints were constructed, each with between one and seven facts replaced. Of the twenty-eight, each subject received six. Table 1 shows the mean coherence score of the degraded viewpoints. Statistical analysis shows a significant difference in the mean coherence of "pure" viewpoints and degraded viewpoints.

A final study adds more evidence that passages vary in coherence based on their source and that viewpoints extracted by the View Retriever are consistently judged to be coherent. A two-way analysis of variance, computed by Paul Cohen[2], determined that there was no significant interaction effect between:

[2]Computer Science Department, University of Massachusetts at Amherst

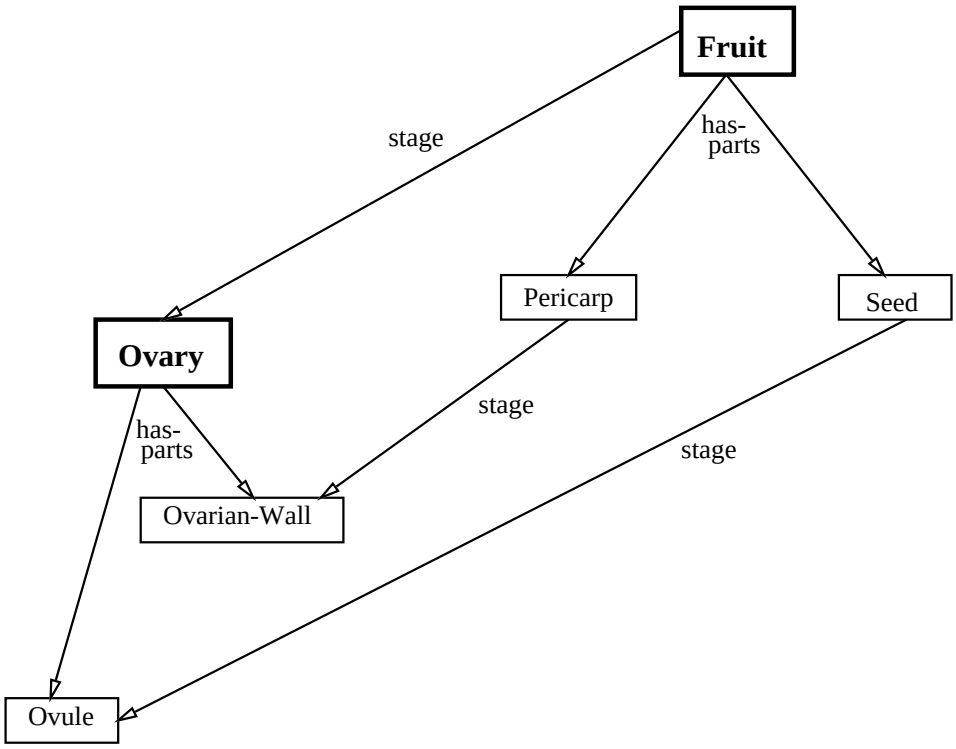


Figure 11: The composite viewpoint that describes the *stage* relations between the parts of the fruit and the parts of the ovary. The View Retriever extracted this viewpoint from the Botany Knowledge Base.

(*composite* (Fruit dimension structural) (Ovary dimension structural) stages)

The composite viewpoint, shown in Figure 11 includes the parts of the fruit (the pericarp and the seed), the parts of the ovary (the ovule and the ovarian wall), and the *stage* relations between them, such as the facts that the ovule is a developmental stage of the seed and the ovarian wall is a developmental stage of the pericarp.

Other examples involve two taxonomic viewpoints put into correspondence. For example, living things participate in reproduction, so the View Retriever can put different specializations of *Reproducing-Structure* into correspondence with the different specializations of *Reproduction* they engage in. Similarly, angiosperms have flowers, so the View Retriever can put different specializations of *Angiosperm* into correspondence with different specializations of *Flower*.

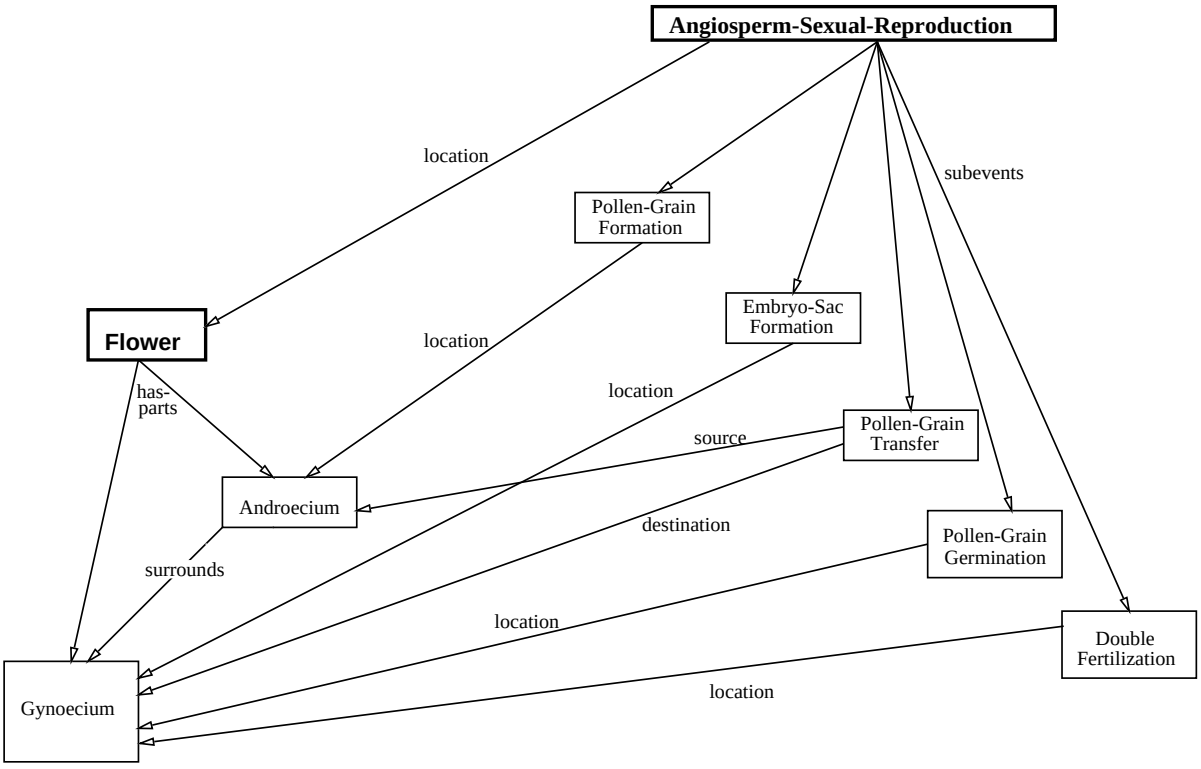


Figure 10: The composite viewpoint that describes the roles of the flower's parts in the subevents of angiosperm reproduction. The View Retriever extracted this viewpoint from the Botany Knowledge Base.

determines the correspondence. For the third type of composite viewpoint, a relation (*part-of, actor-in*, etc.) determines the correspondence. In this type of composite viewpoint, a group of entities from the first viewpoint all have the same kind of relationship to entities from the second viewpoint. For example, the View Retriever can put a structural viewpoint of an object (which describes the object's parts) into correspondence with a procedural viewpoint of an event (which describes the event's steps or subevents) along *actor-in* relations. This correspondence links each of the object's parts to the subevent(s) in which it is an actor of some kind. The resulting viewpoint describes the roles that the object's parts play in the subevents of the event. (Paris's "process strategy" extracts similar descriptions of how the components of a device enable it to perform some function [41, 42]. The View Retriever extracts this and other kinds of composite viewpoints.)

The specification for a composite viewpoint has one of the following forms:

(*composite* ⟨viewpoint1⟩ ⟨viewpoint2⟩ *compare*)
(*composite* ⟨viewpoint1⟩ ⟨viewpoint2⟩ *contrast*)
(*composite* ⟨viewpoint1⟩ ⟨viewpoint2⟩ ⟨slot-name⟩)

where *viewpoint1* and *viewpoint2* are individual viewpoints (or specifications for them). For the third type of composite viewpoint, *slot-name* is a relation that holds between the concepts of interest of *viewpoint1* and *viewpoint2*. It specifies the correspondence to be established between the viewpoints. The remainder of this section focuses on the third type of composite viewpoint.

Consider the composite viewpoint that puts a structural viewpoint of an object into correspondence with a procedural viewpoint of an event along *actor-in* relations. This composite viewpoint, which describes the function of an object (and its parts) in an event (and its subevents), is requested by the following specification:

(*composite* (⟨object⟩ dimension structural) (⟨event⟩ dimension procedural) actor-in)

For example, the viewpoint that describes the roles of a flower's parts in the steps of angiosperm (flowering plant) reproduction is specified as follows:

(*composite* (Flower dimension structural) (Angiosperm-Sexual-Reproduction dimension procedural)
actor-in)

Figure 10 shows the contents of this viewpoint, as extracted from the Botany Knowledge Base by the View Retriever.

The View Retriever extracts this composite viewpoint by the following procedure. First, it extracts the two individual viewpoints (the structural viewpoint of *Flower* and the procedural viewpoint of *Angiosperm-Sexual-Reproduction*). Then, it determines which of the flower parts in the structural viewpoint are related to *Angiosperm-Sexual-Reproduction* or one of its subevents (as given in the procedural viewpoint) by an *actor-in* relation (or some more specific relation, such as *location-of*). The View Retriever omits from the composite viewpoint those parts that are not actors in any subevent of *Angiosperm-Sexual-Reproduction*. For example, *Corolla* (the flower's petals) appears in the structural viewpoint of *Flower*, but the View Retriever excludes it from the composite viewpoint because the corolla doesn't participate in reproduction. Similarly, the View Retriever omits those subevents of *Angiosperm-Sexual-Reproduction* that do not involve any of the parts in the structural viewpoint of *Flower*, such as *Fruit-Ripening*.

Another example of the third type of composite viewpoint is one that describes the parts of a plant ovary as related to the parts of the fruit of which the ovary is a developmental stage. This viewpoint has the following specification:

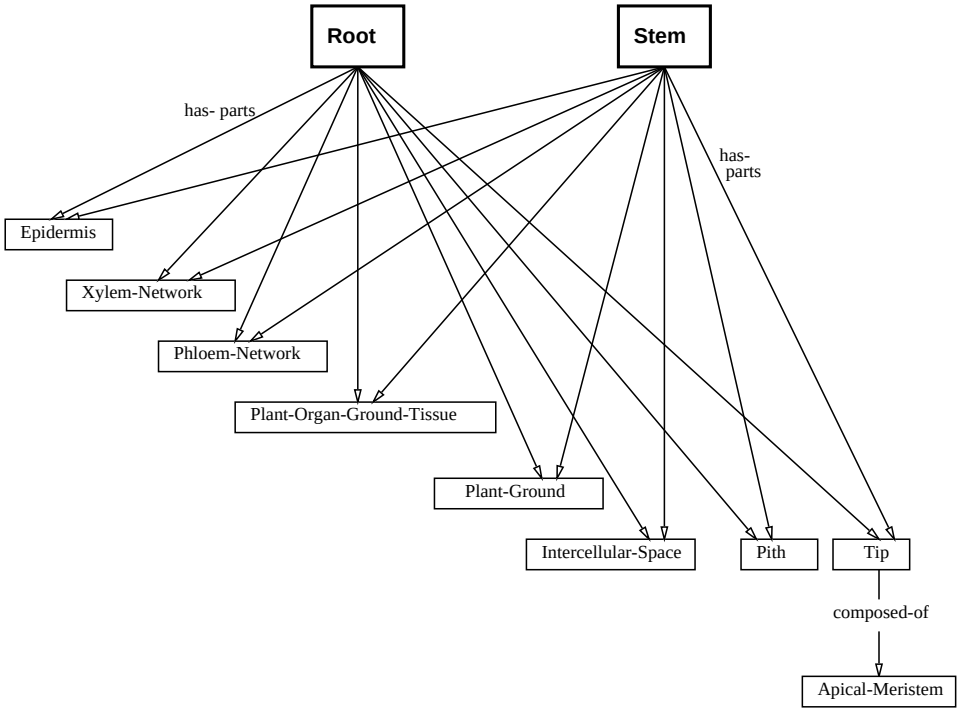


Figure 9: Composite viewpoint that compares the structure of a root to the structure of a stem, as extracted from the the Botany Knowledge Base by the View Retriever.

(Squirrel *as-having* agent-in Seed-Dispersal)

using the stored viewpoint

(Animal *as-having* agent-in Transportation).

If the stored viewpoint contains the fact that animals are usually larger than the things they transport, then the View Retriever would include in the new viewpoint the fact that squirrels are usually larger than the seeds they disperse.

If the concepts of interest of the two viewpoints are *siblings*, as with *Squirrel* and *Bird*, then finding corresponding features is more difficult. It involves finding pairs of features that share a common abstraction, such as ⟨*Bird, mode-of-travel, Flight*⟩ and ⟨*Squirrel, mode-of-travel, Walking*⟩. This matching task is similar to the task of constructing an analogy, given the target and base concepts and the relevant features of the base concept.

If the View Retriever does not find a similar viewpoint in the knowledge base from which to construct the requested viewpoint, then the View Retriever determines what other type of viewpoint includes the feature of interest and returns that viewpoint. This involves determining which basic dimension includes the feature of interest and extracting a viewpoint of the concept of interest along that basic dimension. For example, to extract

(Squirrel *as-having* agent-in Seed-Dispersal)

the View Retriever recognizes that the slot *agent-in* belongs to the *functional* basic dimension (by retrieving the value of (*agent-in slot-dimension*), so it extracts instead the viewpoint

(Squirrel *dimension* Functional),

which includes information about other activities in which squirrels engage. This method of extracting *as-having* viewpoints takes advantage of the inter-relevance of features within the same basic dimension.

3.5 Composite Viewpoints

In addition to extracting individual viewpoints as described above, the View Retriever also extracts composite viewpoints. This involves more than simply concatenating the contents of two individual viewpoints. Rather, it involves putting them into correspondence and removing the portions that do not correspond.

The View Retriever extracts three types of composite viewpoints. The first two are *compare* and *contrast*, wherein the View Retriever highlights the similarities or differences between two concepts under a particular viewpoint. For example, the View Retriever can compare a structural viewpoint of a root to a structural viewpoint of a stem, as shown in Figure 9. To compare two viewpoints, the View Retriever retains only those features that are common to both viewpoints or that share a common abstraction. For example, one part of *Root* is *Root-Intercellular-Space*, and one part of *Stem* is *Stem-Intercellular-Space*, so the View Retriever includes in the comparison the feature *has-part* = *Intercellular-Space*. To contrast two viewpoints, the View Retriever retains only the facts that appear in one viewpoint but not the other. McCoy stresses the importance of adhering to a single type of viewpoint when comparing or contrasting two concepts [32].

Extracting composite viewpoints involves finding the elements of two individual viewpoints that correspond. For comparisons, equality or similarity determines the correspondence. For contrasts, inequality

3.4 *As-having* Viewpoints

An *as-having* viewpoint contains information about a concept that is relevant to some specified feature of the concept. For example, the viewpoint "seed coat as having no chlorophyll" contains facts like "a seed coat is usually not green" and "a seed coat is not photosynthetic."

The specification of an *as-having* viewpoint has the following form:

(⟨concept of interest⟩ *as-having* ⟨slot⟩ ⟨value⟩)

As with other types of viewpoints, the concept of interest can be given by description or by address, and it can be a concept in the actual or virtual knowledge base. The *slot* and *value* in the specification indicate the *feature of interest*, the feature to which facts in the viewpoint must be relevant. For example, the viewpoint "seed coat as having no chlorophyll" is specified by

(Seed-Coat *as-having* percent-chlorophyll Zero)

The ideal method for extracting *as-having* viewpoints is to use a theory of relevance to determine what facts about the concept of interest are most relevant to the feature of interest. The View Retriever would compare each fact known about the concept of interest to the feature of interest using a relevance measure, and it would include in the *as-having* viewpoint only the facts judged most relevant. Unfortunately, a general, prescriptive measure of relevance is not yet available. (Hobbs's coherence relations [19, 20] and Mann and Thompson's rhetorical predicates [29] characterize some of the ways in which one fact is relevant to another, but most of these characterize discourse coherence rather than viewpoint coherence.) Therefore, to select the facts that constitute an *as-having* viewpoint, the View Retriever depends on stored knowledge of relevance. This knowledge takes the form of viewpoints stored in the knowledge base.

To extract an *as-having* viewpoint, the View Retriever first looks for a stored *as-having* viewpoint whose feature of interest is the same as (or more general than) the specified feature of interest, but with a different concept of interest. For example, to extract the following viewpoint:

(Squirrel *as-having* agent-in Seed-Dispersal)

the View Retriever first searches the knowledge base for a similar stored viewpoint, such as one of the following:

- (Mammal *as-having* agent-in Seed-Dispersal)
- (Bird *as-having* agent-in Seed-Dispersal)
- (Animal *as-having* agent-in Transportation)

Note that although the stored viewpoint's *feature* of interest must be either the same as or more general than the specified feature of interest, the stored viewpoint's *concept* of interest need not be related to the specified concept of interest (as with *Squirrel* and *Bird*).

If a similar viewpoint is found in the knowledge base, the View Retriever uses it to determine which facts to include in the new viewpoint. For each feature in the stored viewpoint, the View Retriever looks for a corresponding feature of the specified concept of interest. The way this is done depends on the taxonomic relationship between the concepts of interest of the two viewpoints.

If the stored viewpoint's concept of interest is a generalization of the specified concept of interest, then finding corresponding features involves finding features of the specified concept of interest that specialize features in the stored viewpoint. For example, consider extracting the viewpoint

dimension(s). For example, to request a viewpoint of photosynthesis as a kind of production, but one that only includes information about the actors in photosynthesis (*i.e.,* information within the behavioral basic dimension), the user gives the following viewpoint specification:

(Photosynthesis *as-kind-of* (Production *dimension* Behavioral)

This is the viewpoint actually shown in Figure 6. The complete viewpoint of "photosynthesis *as-kind-of* production" additionally includes information from the procedural and modulatory dimensions, such as the subevents of photosynthesis and what other processes it affects. The viewpoint (Photosynthesis *as-kind-of* Energy-Transduction) shown in Figure 7 is likewise restricted to the behavioral dimension.

3.3.2 Related Work

Viewpoints created by the View Retriever along basic dimensions are similar to the *perspectives* Suthers suggests for explanation generation [55, 56]. Suthers describes a perspective as "an abstract characterization of the kind of knowledge provided by a class of models." Suthers's set of perspectives appears to be a subset of the basic dimensions given here, although he gives only their names: structural, functional, causal, constraint, and process. Suthers uses perspectives to restrict the information selected for an explanation so as to minimize the number of unfamiliar concepts. A concept is considered familiar if its relation to a known concept lies within a chosen perspective.

Viewpoints extracted along basic dimensions are also similar to the *domain perspectives* ROMPER uses to correct users' misconceptions about domain objects [31, 32]. A domain perspective is a group of slots and their *salience values*. They are defined and represented independently of the generalization hierarchy (unlike *as-kind-of* viewpoints). When applied to a domain object, a perspective acts as a filter on its attributes; only the attributes with the highest salience values under that perspective are included in the response. To correct a user's misconception, ROMPER selects an appropriate domain perspective by analyzing the discourse history and the model of the user, then it uses this perspective to select information that is most relevant to the source of the misconception.

ROMPER's perspectives are unlike the basic dimensions given here in that perspectives are specific to the domain of financial securities, but the basic dimensions are generally applicable. The advantage of McCoy's approach is that the slots within a perspective have different degrees of relevance, rather than being simply relevant or irrelevant.

ROMPER does not include a type of perspective analogous to *as-kind-of* viewpoints. Because viewpoints that are tied to the generalization hierarchy (*e.g.*, *as-kind-of* viewpoints) are insufficient to characterize the breadth of viewpoints that people use, and because McCoy's method for generating domain perspectives (and the method given here for basic dimensions) is at least as powerful as a method for generating *as-kind-of* viewpoints, McCoy rejects the latter in favor of the former. The advantage of including a mechanism for generating *as-kind-of* viewpoints, even if redundant, is that *as-kind-of* viewpoints impose fewer demands on the knowledge engineer. The knowledge engineer must explicitly represent the group of slots making up each perspective (and each basic dimension), but the system extracts *as-kind-of* viewpoints using only pre-existing domain knowledge. For this reason, the View Retriever includes methods for generating *as-kind-of* viewpoints. It also includes methods for generating *as-having* viewpoints, as the next section describes.

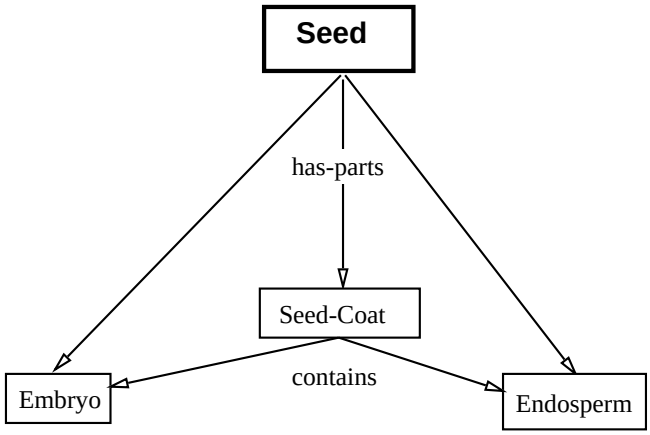


Figure 8: A structural viewpoint of *Seed* as extracted from the Botany Knowledge Base by the View Retriever.

frame is *has-part*, so the View Retriever retrieves the following triples, using traditional frame-slot access methods:

- ⟨*Seed, has-parts, Seed-Coat*⟩,
- ⟨*Seed, has-parts, Embryo*⟩, and
- ⟨*Seed, has-parts, Endosperm*⟩.

Next, the View Retriever selects interconnection relations for the specified basic dimension. For the structural dimension, interconnection relations are *connected-to, contains, surrounds*, etc. The View Retriever looks for these sorts of relationships between the selected parts of the seed (the seed coat, embryo, and endosperm). The View Retriever finds these relationships even if they are not represented directly on the *Seed* frame. The View Retriever finds the following triples:

- ⟨*Seed-Coat, contains, Embryo*⟩
- ⟨*Seed-Coat, contains, Endosperm*⟩

The resulting viewpoint, shown in Figure 8, contains the information that the seed is made up of a seed coat containing an embryo and an endosperm.

3.3.1 Combining Basic Dimensions with *As-kind-of* Viewpoints

As the previous section mentioned, basic dimensions can be combined with *as-kind-of* viewpoints to enhance their coherence. To do so, the user requests an *as-kind-of* viewpoint as before, except that instead of specifying a reference concept, the user specifies a *viewpoint* of the reference concept along some basic

This set of basic dimensions is designed for domains concerned with physical objects and processes. It is a compilation of knowledge types drawn from several sources, including Lakoff and Johnson [24], instructional text analyses [52], and large knowledge-engineering efforts [25, 45]. The list must be extended to reflect the kinds of knowledge found in other kinds of domains and to reflect types of knowledge specific to a particular domain. For example, Lakoff and Johnson suggest two basic dimensions for human artifacts in addition to those given above for objects: *purposive*, which includes information about the human goals that the object was designed to satisfy, and *motor-activity*, which includes information about how people use or interact with the object. For human activities, they suggest the *purpose* dimension in addition to those given above for processes.

To extract viewpoints along basic dimensions, the View Retriever requires knowledge of which slots in the knowledge base are within each dimension. Experience with our knowledge base indicates that this knowledge is easily represented directly in the knowledge base, because the distinctions the basic dimensions make also occur in the slot hierarchy, and because it is usually appropriate for a slot to inherit the basic dimension(s) of its generalizations. In the Botany Knowledge Base, each frame that represents a slot has a *slot-dimension* facet to indicate which basic dimension(s) the slot belongs to. If no value is specified for *slot-dimension*, then the slot inherits the *slot-dimension* of more general slots. Most slots belong to exactly one basic dimension, but this is not required by the View Retriever.

Representing knowledge of basic dimensions directly in the knowledge base has two advantages. First, the set of basic dimensions is easily refined and extended. Second, the View Retriever's algorithm for extracting viewpoints is independent of the particular set of basic dimensions used.

The specification for a viewpoint extracted along a basic dimension has two required parameters, the concept of interest and the name of the basic dimension. For example, a structural viewpoint of *Seed* has the following specification:

(Seed *dimension* structural)

As with *as-kind-of* viewpoints, the concept of interest can be given by description or by address, and it can be a concept in the actual or virtual knowledge base.

Some basic dimensions have as optional parameters *relation restrictions* and *value restrictions*. When the specification includes a relation restriction, the View Retriever extracts a viewpoint that includes only triples involving slots that are specializations of (or equal to) the given relation. For example,

(Chloroplast *dimension* functional (*relation-restriction* producer))

specifies a viewpoint that includes only information about processes in which a chloroplast is the producer. When the specification includes a value restriction, the View Retriever extracts a viewpoint that includes only triples whose values are specializations of (or equal to) the given concept. For example,

(Xylem *dimension* functional (*value-restriction* Transportation))

specifies a viewpoint that includes only information about transportation processes involving the xylem (the conduit for water flow in a plant).

The View Retriever extracts a viewpoint along a basic dimension first by retrieving facts about the concept of interest that belong to the basic dimension. For example, to extract a structural viewpoint of a plant seed, specified by (Seed *dimension* Structural), the View Retriever first retrieves from the *Seed* frame the values of all slots that belong to the structural dimension. A structural slot that appears on the *Seed*

- **Spatial-Superstructural**, which includes the object(s) of which the object is a part, the other sibling parts, and the connections and spatial relations among them.
- **Perceptual**, which includes information regarding how people perceive (see, hear, etc.) the object. This dimension includes the shape, symmetry, size, color, and temperature of the object.
- **Functional**, which includes what the object "does" (the processes in which it is an actor). The **Active-Functional** dimension is a subtype of the functional dimension that includes only information about processes in which the object is an active, rather than a passive, actor. (For example, the producer in a production process is an active actor, but the location of the process is a passive actor.)
- **Temporal**, which includes the temporal parts of an object (its stages or states). It also includes, as interconnection relations, the temporal ordering constraints among the stages or states.
- **Temporal-Superstructural**, which includes the objects of which this object is a stage or state, the other sibling stages/states, and their temporal ordering constraints.

Basic dimensions for processes:

- **Behavioral**, which includes the types and roles of the actors in the process and the changes that the process effects upon them. The behavioral dimension also includes initial and final conditions of the process, the relative amounts/sizes of the actors, and the forms of the actors.
- **Procedural**, which includes the steps (subevents) of the process and, as interconnection relations, the temporal ordering constraints among the steps.
- **Event-Superstructural**, which includes the process(es) of which the process is a step, the other sibling steps, and the temporal ordering constraints among them.

Basic dimensions for both objects and processes:

- **Taxonomic**, which includes the subcategories (specializations) of a category, the relative sizes of the subcategories, the criteria for the breakdown, and, as interconnection relations, information about which subcategories are disjoint.
- **Taxonomic-Superstructural**, which includes the generalization(s) of the given category, other categories that share the same generalization(s), their relative sizes, and disjointness relations among them.
- **Modulatory**, which includes information about how one object or process affects other objects or processes. This dimension includes causal relationships (*e.g.* causes, enables, prevents, facilitates), which constitute the subtypes **Causal-Agent** and **Causal-Recipient**, and qualitative influences [11, 23] between quantities (*e.g.* directly-affects, inversely-influences, correlated-with), which constitute the subtypes **Influence-Agent** and **Influence-Recipient**. The latter subtypes include information about the relative strengths of influences, the conditions under which they hold, and their saturation points.

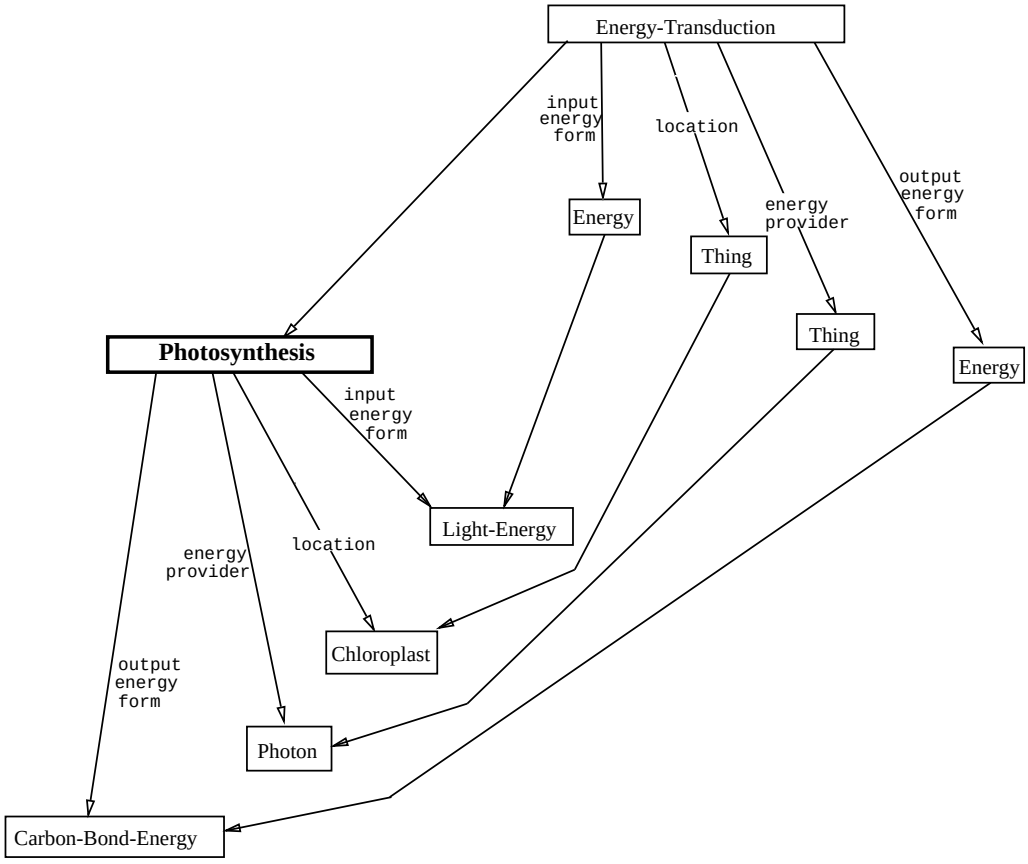


Figure 7: The viewpoint of "photosynthesis *as-kind-of* energy transduction" as extracted from the Botany Knowledge Base by the View Retriever.

When extracting a viewpoint of a concept, the View Retriever filters out many facts about the concept that are irrelevant to the viewpoint. *As-kind-of* viewpoints provide two kinds of filtering. The first filter removes redundant features, those that the concept of interest and the reference concept have in common. For example, the View Retriever excludes from the "photosynthesis *as-kind-of* production" viewpoint the fact that photosynthesis has a temporal duration, because this is true of any production event, and the viewpoint contains the fact that photosynthesis is a kind of production. (If, on the other hand, *Photosynthesis* had a more specific value for *duration* than *Production* has, then the View Retriever would include *duration* in the viewpoint.) The second filter removes irrelevant features, features of the concept of interest that do not fit within the conceptual structure of the reference concept. For example, although the knowledge base contains the information that photosynthesis converts light energy into carbon bond energy, the View Retriever excludes this information because it does not fit within the conceptual structure of production, as represented in the Botany Knowledge Base. (That is, *Production* is not within the domain of slots *input-energy-form* and *output-energy-form*.) The View Retriever *does*, however, include that information in the viewpoint "photosynthesis *as-kind-of* energy transduction," shown in Figure 7.

The central notion behind *as-kind-of* viewpoints is that one can emphasize different aspects of a concept by differentiating it with respect to different generalizations. This, of course, is not a novel idea; it has appeared in a variety of disciplines since the time of Aristotle [3]. In the field of artificial intelligence, it appeared as early as 1977 as the basis of KRL, one of the first frame-based representation languages [5]. More recently, McKeown and Suthers have designed systems that automatically extract concept descriptions of this sort [33, 34, 56].

The problem with *as-kind-of* viewpoints that the View Retriever extracts (and with the descriptions McKeown's and Suthers's systems extract, when applied to a large knowledge base) is that, even though they focus on a single aspect of a concept, they are nonetheless too unconstrained. They often mix several different kinds of information. For example, the complete viewpoint "pine tree *as-kind-of* tree" includes facts about how a pine tree looks different from a prototypical tree, how its internal structure is different, how its development differs, how its physiology differs, etc.

As a solution, this research includes an additional, orthogonal method of structuring concepts, one that the View Retriever can combine with the method for generating *as-kind-of* viewpoints to further constrain their contents. The next section discusses this method of generating viewpoints and how the View Retriever uses it to enhance the coherence of *as-kind-of* viewpoints.

3.3 Viewpoints Constructed Along Basic Dimensions

Lakoff and Johnson say that people structure their concepts in two ways, by relating them to other concepts (as with *as-kind-of* viewpoints) and according to *basic dimensions* [24]. Basic dimensions are general types of facts, such as facts about an object's structure, function, or appearance or facts about a process's actors or steps. Facts within the same basic dimension convey similar kinds of information. The View Retriever constructs viewpoints along the following basic dimensions for objects and processes:

Basic dimensions for objects:

- **Structural**, which includes the parts or substances that make up the object and their relative sizes and numbers. It also includes the connections and spatial relations among them, called *interconnection relations*.

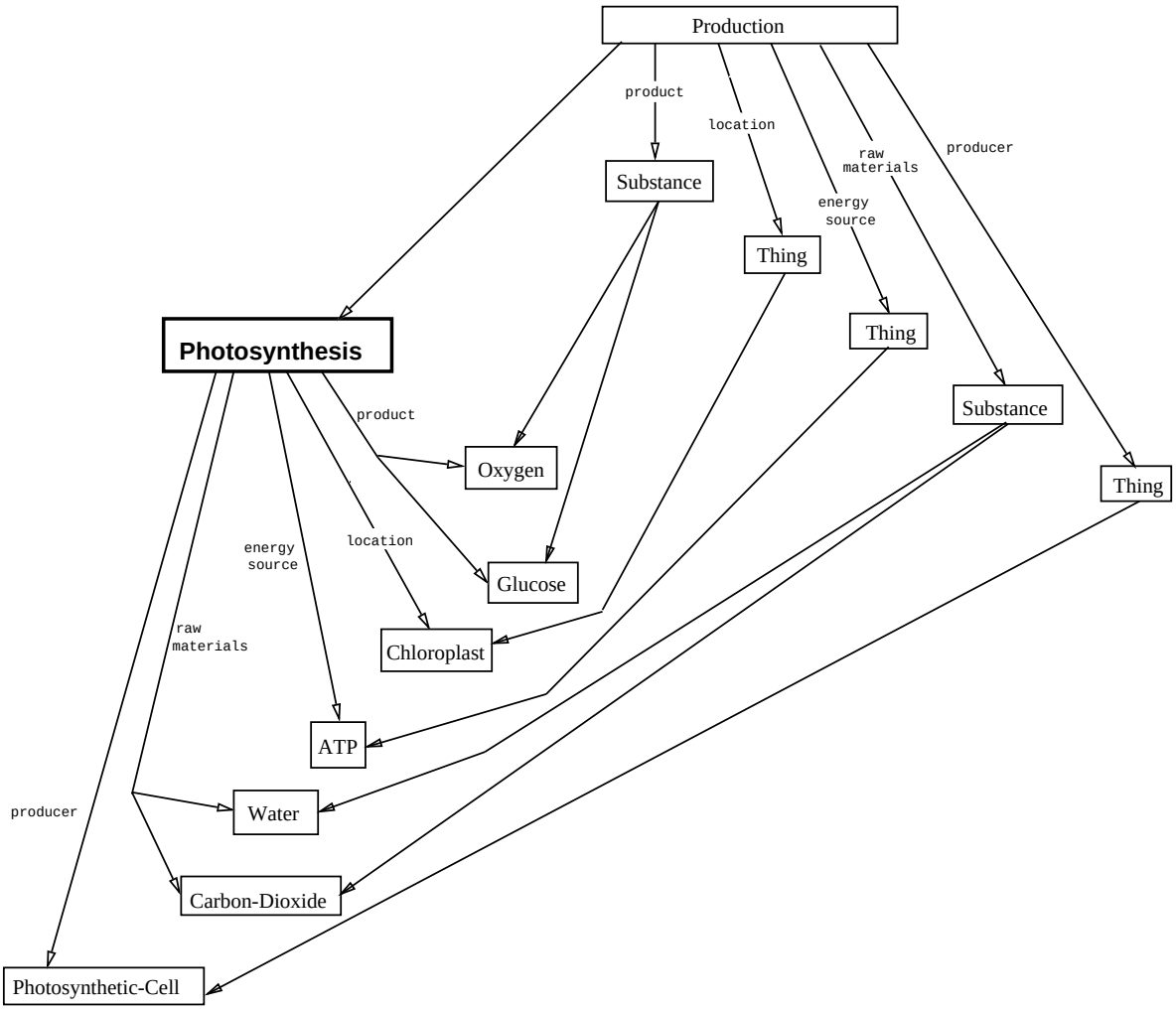


Figure 6: The viewpoint of "photosynthesis *as-kind-of* production" as extracted from the Botany Knowledge Base by the View Retriever.

A user specifies a viewpoint by indicating the type of viewpoint required and the concept of interest. The concept of interest can be specified by description or by frame address, and it can be a concept in the actual or virtual knowledge base.

A second major contribution of this work is a collection of methods for dynamically extracting viewpoints from large knowledge bases. The following sections describe the methods the View Retriever uses to extract viewpoints of the four types given above.

3.2 *As-kind-of* Viewpoints

An *as-kind-of* viewpoint describes a concept in terms of a more general concept. For example, the viewpoint "photosynthesis *as-kind-of* production" consists of those facts that explain how photosynthesis is a special kind of production, such as what its raw materials and products are. Figure 6 shows a portion of this viewpoint as produced by the View Retriever.

The specification of an *as-kind-of* viewpoint requires two parameters:

- the *concept of interest*, the concept that is the focus of the viewpoint, and
- the *reference concept*, a generalization of the concept of interest (although not necessarily an immediate generalization).

For example, the viewpoint shown in Figure 6 has the following specification:

(Photosynthesis *as-kind-of* Production).

The View Retriever extracts *as-kind-of* viewpoints by first selecting relevant facts about the concept of interest (*i.e.,* triples of the form ⟨*concept-of-interest, slot, value*⟩). A triple is considered relevant if some more general triple appears on the frame for the reference concept. The triple ⟨*reference-concept, slot′, value′*⟩ is more general than ⟨*concept-of-interest, slot, value*⟩ if both of the following conditions hold:

1. *value* = *value′* or *value* is a specialization of *value′*.
2. *slot* = *slot′* or *slot* is a specialization of *slot′* and the *domain* of *slot* is some concept that is a specialization of the reference concept and a generalization of the concept of interest.

For example, the viewpoint shown in Figure 6 contains the fact that photosynthesis produces glucose, because it is known that production processes typically produce some substance and glucose is a special kind of substance. More specifically, the View Retriever includes ⟨*Photosynthesis, product, Glucose*⟩ in the viewpoint "photosynthesis *as-kind-of* production" because the knowledge base contains ⟨*Production, product, Substance*⟩, which is more general than ⟨*Photosynthesis, product, Glucose*⟩ because *Substance* is a generalization of *Glucose* [condition (1) above].

After the View Retriever selects relevant facts involving the concept of interest, it adds to the viewpoint the connections between these facts and the more general facts involving the reference concept. For example, the viewpoint in Figure 6 includes not only the fact that photosynthesis produces glucose, but also the facts that photosynthesis is a kind of production, that production processes produce some substance(s), and that glucose is a kind of substance. These connections provide the justification for the viewpoint's contents and are sometimes useful for application programs. For example, a tutoring system can use them to relate new information in an explanation to the student's background knowledge.

that textual coherence involves "making sure an object is named before it is described, or making sure that if an object's components are mentioned, they are mentioned in one place, not scattered about the text" [49].

The third factor contributing to coherence is *contextual coherence*, the degree to which the facts in an explanation are related to the context in which they are presented. For example, *discourse coherence* reflects the degree to which new information relates to earlier passages in a discourse. Hobbs [19, 20] and Rhetorical Structure Theory [29] propose *coherence relations* and *rhetorical relations* as the "legal moves" connecting portions of a coherent discourse. Similarly, *global coherence* [19] is the degree to which the utterances in a discourse pertain to the speaker's specific communication plan [14, 35, 36].

Viewpoint coherence is the focus of this research. Because it is determined entirely by the associations among the facts comprising an explanation, viewpoint coherence is typically a precursor for the other factors contributing to coherence. Even with this restricted focus, coherence remains an ill-defined notion. The goal of this work, however, is not to provide a precise, operational *definition* of coherence, but to understand how an access method can generate viewpoints that will be judged coherent by people. This is the basis for the empirical evaluation of our access method, as described in section 3.6.

3.1 Task Description

The task of extracting viewpoints can be described as follows:

Given:

- a knowledge base containing declarative knowledge of domain concepts, and
- a viewpoint specification, which indicates the type of viewpoint required and the concept of which the viewpoint will be taken (the *concept of interest*).

Return: A collection of facts (⟨*frame, slot, value*⟩ triples) from the knowledge base that constitutes the specified viewpoint. This collection includes both facts that are explicit in the knowledge base as well as facts that must be computed (*i.e.,* facts that are in the virtual knowledge base).

Our methods for accessing viewpoints are implemented in a component of KASTL called the *View Retriever* (a term proposed by Suthers [54]). The View Retriever is currently used with our Botany Knowledge Base, but it is designed to work for any physical domain and to be easily extended to work in nonphysical domains, such as those involving abstract concepts or mental processes.

A major contribution of this work is a framework of viewpoint types that are independent of any domain and task. The current framework of viewpoint types includes

- *as-kind-of* viewpoints, which describe the concept of interest by relating it to a more general concept.
- viewpoints constructed along *basic dimensions*, which describe particular kinds of features of the concept of interest (structural features, functional features, etc.).
- *as-having* viewpoints, which include features about the concept of interest that are relevant to a user-specified feature of the concept.
- *composite* viewpoints, which combine, and put into correspondence, two individual viewpoints.

access operations.

The cost of classifying a new concept, then, is the product of

- the number of frames to be compared with the input description, and
- the number of frame-slot accesses required to compare a particular frame with the input description (and compute the subsumption relationship between them).

Because this research deals with accessing large knowledge bases, the above cost analysis has focused on the cost of classification as a function of the number of frames in the knowledge base. In this way, we can determine how classification scales up as the knowledge base grows. Woods's theoretical analysis, corroborated by the empirical evidence given here, indicates that, as a function of the number of frames in the knowledge base to be compared with the input description, the complexity of KASTL's classification algorithm is logarithmic for typical inputs and linear in the worst case. For the current size of the Botany Knowledge Base and the 53 concepts selected from a biology textbook, the number of frames to be compared to the input description to classify the concept averaged 4.3%.

3 Extracting Viewpoints from Large Knowledge Bases

After our access methods locate a concept, they enable users to extract viewpoints of it. A viewpoint is a coherent collection of facts (*i.e.,* $\langle$*frame, slot, value*$\rangle$ triples) that describes a concept from a particular perspective. For example, a structural viewpoint of the concept *Seed-Coat* describes the substances and parts that make up a seed coat and how they are connected. The viewpoint of *Seed-Coat* as a kind of *Container* includes information about what parts of the seed the seed coat contains, whether the seed coat has openings, etc. Each concept has multiple (possibly overlapping) viewpoints, which give different perspectives of the concept.

Extracting viewpoints from the potentially large set of facts about each concept is important for virtually all users of large knowledge bases, as discussed further in section 3.3.2. Briefly, many explanation-generation systems require viewpoints to produce explanations that are complete and coherent [56, 34, 27, 32, 35]. Qualitative modeling systems use viewpoints to increase efficiency and to make consistent modeling assumptions (*e.g.,* the *perspectives* of [10], the *views* of [11], and the *ontological perspectives* of [28].)

The coherence of a set of facts is determined by three factors. The first factor is *viewpoint coherence*, the degree to which the facts are interrelated by their contents. For example, all the facts in a coherent viewpoint of "photosynthesis as a carbon dioxide utilization process" must pertain to the fact that photosynthesis utilizes carbon dioxide. Hobbs points out that viewpoint coherence is not simply sharing a common referent [19]. That is, a set of facts that describe the same concept is not necessarily coherent. Consider the following example from [19]:

> Ronald Reagan used to act in cowboy movies. He appointed Caspar Weinberger as Secretary of Defense.

Although the statements are *cohesive*, because they both refer to Ronald Reagan [17, 12, 9], they are not coherent. Viewpoint coherence requires a stronger relationship between facts.

The second factor contributing to the coherence of a set of facts is *textual coherence*, the degree to which a collection of facts is well organized. Sibun, who defines coherence in terms of these first two factors, says

concept description being classified. The size of the taxonomy has a much greater impact on the overall cost of classification, because description size is typically small, but taxonomy size may be quite large.

To address the limitations of past research, Woods has analyzed the complexity of classification based on intensional subsumption [58]. Woods shows that the complexity of classification, as a function of how many of the frames in the taxonomy are compared to the input concept description, is logarithmic or less for typical inputs and linear in the worst case.

Woods uses three parameters to capture how the characteristics of the knowledge base affect the cost of classification. The first is the downward branching ratio, r, the average number of immediate specializations for concepts that are not leaves of the taxonomy. The second is a parameter B, which reflects how "out of balance" the taxonomy is. The third is a parameter W, the width of the ancestor chain above a typical concept due to multiple generalizations. Woods estimates that B and W are in the range of one to three for most knowledge bases.

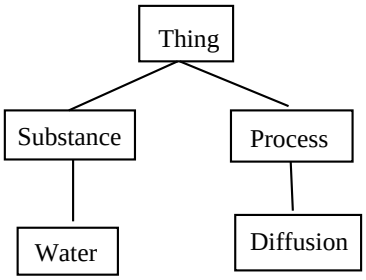
The cost-dominant steps of KASTL's classification algorithm (shown in Figure 3) are steps 1, 2, and 3. For steps 1 and 2, which find the most specific generalizations of the new concept, Woods estimates the maximum cost for typical inputs is $rBWlogN$ frames compared to the input description, where N is the number of frames in the knowledge base. For step 3, which finds the most general specializations of the new concept, Woods estimates that, for typical inputs, $r(r+1)$ frames must be compared to the input description. Thus the total number of comparisons is $rBWlogN + r(r+1)$ frames. The value of r (the downward branching ratio) for the Botany Knowledge Base is 4. Using this value of r and an estimate of 2 for parameters B and W, as suggested by Woods, yields an estimated maximum cost of $16logN + 20$ comparisons for typical inputs. For the Botany Knowledge Base, this would mean comparing a typical input description with at most 5.6% of the 2,665 frames in the knowledge base.

To empirically evaluate the above cost estimate based on Woods's analysis, 53 botanical concepts that are not explicitly represented in the Botany Knowledge Base, such as "nucleus of an epidermal cell" and "tree that grows in a swamp," were selected. These concepts were chosen at random from a biology textbook, so presumably the sample is representative of the domain. The evaluation consisted of measuring, for each concept, the maximum number of frames that KASTL would compare with the input description to classify the concept in the Botany Knowledge Base taxonomy. The average was 117 of the 2,665 total frames, about 4.3%, not significantly different from Woods's estimate. The minimum was 32 frames (1%), and the maximum was 804 frames (30%). (Recall that in the worst case, KASTL would have to examine 100% of the frames in the taxonomy.) The cost of classifying a particular concept depends largely on the position of the concept in the taxonomy. The more general the concept, (*i.e.,* the closer the concept is to the root of the taxonomy), the more costly it is to classify.

The above analysis determined the number of frames to be examined when classifying a new concept. For each frame examined, KASTL must determine the subsumption relationship between the concept the frame represents and the input concept description. Woods estimates the typical-case cost of determining intensional subsumption as $(2m^2 + p^2)$ frame-slot access operations, where m is the number of features in the input concept description and p is the number of base concepts in the input description. [1]

Of 155 concept descriptions found in a chapter of a biology textbook, all but four had a single base concept (*i.e.,* $p \approx 1$ on average), and the average number of features modifying the base concept was 1.3. Thus, the average cost of computing intensional subsumption for these concepts would be 4.4 frame-slot

[1] Woods's estimate is only for non-disjunctive concept descriptions and for those not involving binding constraints among variables.



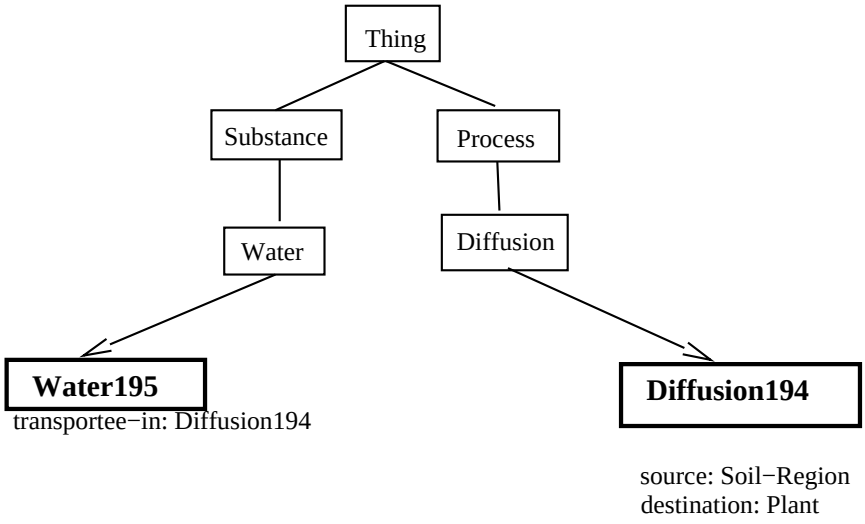


Figure 5: Snapshots of the knowledge base before and after reification of *Water195*, described by (Water (transportee-in (Diffusion (source Soil-Region) (destination Plant)))).

terminates. The only specialization found is *Oxygen-of-Leaf-Photosynthesis*, which will be installed as an immediate specialization of the new concept.

Step 4 of the procedure is to create a frame to represent the new concept (*e.g.*, *Oxygen′*) and reorganize the knowledge base to accommodate it. Reorganization involves installing generalization relations from the new frame to the frames found in steps 1 and 2, installing specialization relations from the new frame to the frames found in step 3, and installing each feature given in the concept description on the new frame. It also involves removing taxonomic relations that become redundant after installing the new taxonomic relations. The bottom portion of Figure 4 shows the result of this reorganization for the knowledge-base fragment shown in the top portion of Figure 4, following the creation of a new frame for the concept described by (Oxygen (end-product-of Photosynthesis)). The specialization relation from *Biologically-Produced-Oxygen* to *Oxygen-of-Leaf-Photosynthesis* is removed, because it is redundant given the new specialization relations from *Biologically-Produced-Oxygen* to *Oxygen′* and from *Oxygen′* to *Oxygen-of-Leaf-Photosynthesis*. Similarly, the redundant specialization relation from *Product-of-Photosynthesis* to *Oxygen-of-Leaf-Photosynthesis* is removed.

The final step in reifying concepts in the virtual knowledge base is to infer new information about the concept and install it on the new frame. This can be done using standard inference methods such as induction from specializations or instances, deduction from rules, or inheritance. This step can be done as the system installs the new frame in the knowledge base, or it can be done on demand as users request slot values. KASTL takes the latter approach.

As a final example, when KASTL is given the request

(Water (transportee-in (Diffusion (source Soil-Region) (destination Plant)))),

which describes "Water moved by diffusion from a soil region to a plant," and the knowledge-base fragment shown in the top portion of Figure 5, KASTL modifies the knowledge base as shown in the bottom portion of Figure 5. In addition to creating a frame to represent the new specialization of *Water*, KASTL also creates a frame to represent the new specialization of *Diffusion*, "diffusion of water from a soil region to a plant," referenced by the nested description.

2.3 Cost Analysis

This section presents a cost analysis of automatic classification, adding new concepts to an existing taxonomy. Automatic classification is the primary and most costly activity in accessing concepts in the virtual knowledge base. This analysis compares the estimated costs computed by Woods [58] with actual costs of adding new concepts to our Botany Knowledge Base [45]. Our knowledge base currently contains 2,665 frames and over 28,000 facts about plant anatomy and physiology.

Previous complexity analyses have four limitations [58]. First, most past research has focused only on the cost of subsumption, rather than the cost of classification. Subsumption, determining whether one concept is more general than another, is but one step in classification; more important is the complexity of the overall classification task. Second, past research has analyzed the cost of determining *extensional subsumption*. As described in Section 2.2.1, KASTL uses a classification algorithm based on *intensional subsumption* rather than extensional subsumption. Intensional subsumption is tractable for a much wider range of representation language expressiveness. The third limitation of past results is that most of them give only worst-case complexity analyses. Users of the knowledge base are often more interested in average-case predictions. The fourth limitation is that most complexity analyses are given in terms of the size of the

thesis), so KASTL repeats the search from *Biologically-Produced-Oxygen*. None of the specializations of *Biologically-Produced-Oxygen* can be pursued further, however, because none of them match or subsume the given concept description. Thus, the only failure point is *Biologically-Produced-Oxygen*. After failing to find a match for the specified concept, KASTL proceeds to create it. KASTL recalls the single failure point, *Biologically-Produced-Oxygen*, to be installed as an immediate generalization of the new concept. *Biologically-Produced-Oxygen* is guaranteed to be the only immediate generalization of the new concept in the portion of the taxonomy rooted at the base concept, *Oxygen*.

Step 2 of the procedure finds the generalizations of the new concept that are *outside* the portion of the taxonomy rooted at the base concept. Although the search for an exact match for a concept description can be limited to the portion of the taxonomy rooted at the base concept, the search for generalizations of the described concept must originate at the root of the taxonomy. Because KASTL is searching for maximally *specific* generalizations only, it need not examine ancestors of the base concept. (The base concept subsumes the described concept by construction, and it is more specific than any of its generalizations.) Specializations of ancestors of the base concept, however, must be examined. For example, although the ancestors of *Oxygen* (*Substance* and *Thing*) will not be maximally specific generalizations of the concept described by (Oxygen (end-product-of Photosynthesis)), the immediate specializations of *Substance* and *Thing* (*Soil, Product-of-Photosynthesis, Object, Process*, and *Slot*) must be examined. If any of these concepts subsumes the given description, then KASTL must also examine its specializations.

KASTL continues searching the taxonomy in this way until it finds no more concepts that subsume the input description. The most specific concepts found will be immediate generalizations of the newly created concept. For example, although none of *Soil, Object, Process*, or *Slot* subsumes the concept described by (Oxygen (end-product-of Photosynthesis)), *Product-of-Photosynthesis* does subsume it. Because *Product-of-Photosynthesis* has no explicit specializations, it is a maximally specific generalization of the given description, and KASTL will install it as an immediate generalization of the new concept.

Step 3 of the procedure is to find the immediate specializations of the concept to be created. This involves finding the most general concepts in the knowledge base that the given concept description subsumes. (The given description subsumes a concept in the knowledge base if every feature in the description is necessary for the concept or generalizes some feature that is necessary.) The search for specializations can originate with any of the generalizations found in steps 1 and 2, preferably the one with the fewest specializations. KASTL chooses a starting concept and examines its immediate specializations. If a concept is subsumed by the given concept description, then KASTL retains it as a maximally general specialization. (Its specializations need not be examined because KASTL is searching for the maximally *general* specializations.) If a concept is *not* subsumed by the given description, then KASTL must also examine all of its specializations. In the worst case, the search continues to the most specific concepts in that portion of the taxonomy. The most general specializations found will be immediate specializations of the new concept.

Recall that for the hypothetical knowledge-base fragment shown in the bottom portion of Figure 4, the most specific generalizations of the concept description (Oxygen (end-product-of Photosynthesis)) are *Product-of-Photosynthesis* and *Biologically-Produced-Oxygen*. Assume that KASTL chooses *Biologically-Produced-Oxygen* as the starting point of the search for specializations. The first specialization of *Biologically-Produced-Oxygen*, *Oxygen-of-Leaf-Photosynthesis*, is subsumed by the given description, because *Leaf-Photosynthesis* is a specialization of *Photosynthesis*. Thus, *Oxygen-of-Leaf-Photosynthesis* is a maximally general specialization and its specializations (if it had any) would not be examined. The second specialization of *Biologically-Produced-Oxygen*, *Respired-Oxygen*, is not subsumed by the given description, thus KASTL must also examine its specializations. In this example, *Respired-Oxygen* has no specializations, so the search

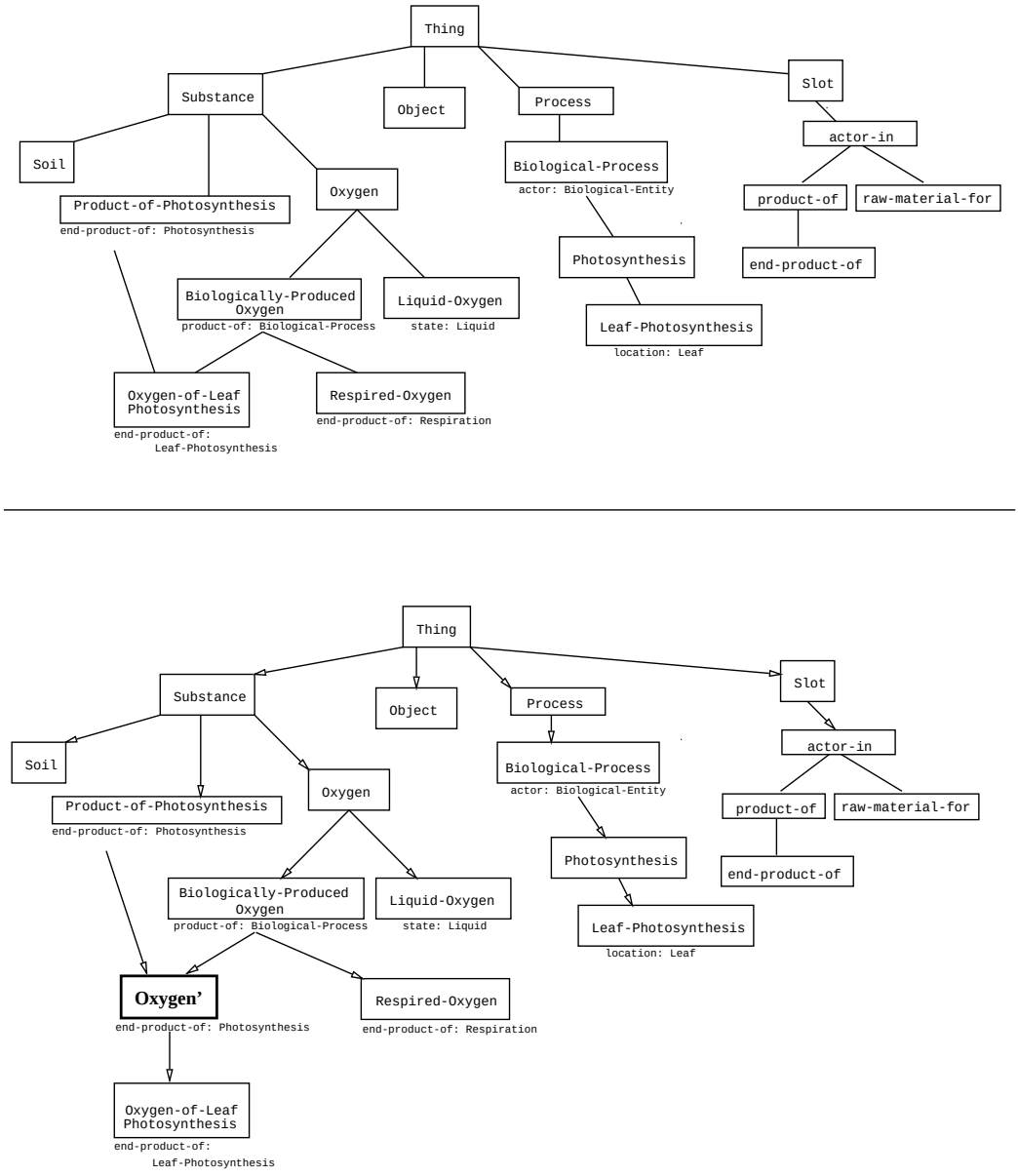


Figure 4: Knowledge-base fragments before and after reorganization to accommodate the new concept *Oxygen′*, described by (Oxygen (end-product-of Photosynthesis)). Only the relevant portions are shown.

1. Find immediate generalizations of the new concept. Recall the failure points of the search for a match. (Figure 1 gives the algorithm for this search.) These failure points are the most specific concepts that are more specific than the base concept but more general than the new concept.
2. Find additional immediate generalizations of the new concept. Find the most specific concepts that are both
   - neither more general nor more specific than the base concept, and
   - more general than the given description.
3. Find the immediate specializations of the new concept. Choose one of the generalizations G found in step 1 or 2, and find the most general concepts that are specializations of G and that are more specific than the given description.
4. Create a new frame.
   (a) Install generalization relations to concepts found in steps 1 and 2.
   (b) Install specialization relations to concepts found in step 3.
   (c) Assert on the new frame each feature given in the input description.
   (d) Remove redundant taxonomic relations.
5. Infer new features for the new concept (optional).

Figure 3: KASTL's procedure for accessing concepts in the virtual knowledge base.

carries no assertional import [7, 6]. Extensional subsumption cannot always be computed solely from terminological or definitional knowledge (such as the information in KRYPTON's TBox [6]). For example, if "Triangle" is defined as "Polygon with three angles," determining that "Polygon with three or more sides" subsumes "Triangle" requires the fact that "every angle of a polygon has a corresponding side," knowledge that is strictly assertional rather than definitional (and hence would appear in KRYPTON's ABox rather than its TBox).

If subsumption is to be computed using only definitional knowledge, a new criterion for subsumption must be used, such as one that is based on concept intensions rather than extensions. Woods introduces such a criterion, called *intensional subsumption* [58]. Intensional subsumption means that the definition (intension) of the subsuming concept is more general than the definition of the subsumed concept. Definition $D1$ is more general than definition $D2$ when every sufficient feature of $D1$ is necessary for $D2$ or generalizes some feature that is necessary for $D2$. For example, under intensional subsumption, "Person whose children are professionals" subsumes "Woman whose children are doctors" (assuming that "Woman" is defined as a kind of "Person" and "Doctor" is defined as a kind of "Professional"), but "Polygon with three or more sides" does not subsume "Polygon with three angles."

To overcome the limitations of traditional classifiers, KASTL's classification algorithm is based on intensional subsumption rather than extensional subsumption. Although Woods proposes intensional subsumption as a more tractable alternative to extensional subsumption, he retains extensional subsumption as the criterion of completeness. That is, Woods says that intensional subsumption should entail extensional subsumption, and that, all else being equal, it is desirable to be as complete as possible with respect to extensional subsumption. This work, by contrast, rejects extensional subsumption in favor of intensional subsumption because extensional subsumption has the undesirable property that two concepts that have empty extensions in all possible worlds are considered to subsume one another (*i.e.,* to be equivalent). Under intensional subsumption, concepts are equivalent only when they have identical intensions.

2.2.2 KASTL's Approach to Accessing Concepts in the Virtual Knowledge Base

Figure 3 gives the procedure KASTL uses to access concepts in the virtual knowledge base. The first two steps of the procedure find the immediate generalizations of the concept to be created. This involves finding the most specific concepts in the knowledge base that subsume the given concept description. (Recall that a concept in the knowledge base subsumes the given description if it has at least one sufficient feature, and each such feature appears in the description or generalizes some feature in the description.) Step 1 takes advantage of the fact that KASTL creates a new concept only after having failed to find the concept in the knowledge base. If KASTL fails to find an exact match for the given concept description, it terminates its search after encountering the most specific specializations of the base concept that are more general than the given concept description. These concepts will be immediate generalizations of the concept to be created. Thus, KASTL avoids repeating the search for these generalizations by recording the failure points of the search for a match.

For example, consider the concept description (Oxygen (end-product-of Photosynthesis)) and the knowledge-base fragment shown in the top portion of Figure 4. To search the knowledge base for a concept matching the description, KASTL begins at the base concept, *Oxygen*. One specialization of *Oxygen*, *Liquid-Oxygen*, neither matches nor is more general than the given description, so KASTL does not pursue it further. The other specialization, *Biologically-Produced-Oxygen*, is more general than the given description (because *product-of* is more general than *end-product-of*, and *Biological-Process* is more general than *Photosyn-*

- The name of the frame created to represent the described concept, and
- A new knowledge base reorganized to accommodate the new frame.

For example, given the concept description (Oxygen (end-product-of Photosynthesis)), the task is to create a frame representing "Oxygen produced by photosynthesis" and to modify the taxonomy to include that frame.

The same concept description language used to provide content addressability is also used to provide access to concepts in the virtual knowledge base. The semantics of the language is the same for both uses; the base concept is a generalization of the described concept, features modifying the base concept are jointly necessary and sufficient, and nested descriptions are matched or created from the inside out. Using the same concept description language for both content addressability and accessing concepts in the virtual knowledge base allows a single user interface. In this way, users do not need to know whether they are accessing existing concepts or virtual concepts.

2.2.1 Related Work

Providing access to concepts in the virtual knowledge base is essentially an *automatic classification* task. Automatic classification involves inserting a new concept into a taxonomy so that it is directly linked to the most specific concepts that subsume it and to the most general concepts that it subsumes [59]. Automatic classification originated with KL-ONE [7] and continued with most of KL-ONE's successors, including KRYPTON [6].

As implemented in KL-ONE, KRYPTON, and their descendants, automatic classification has several limitations. First, many of these languages, including KRYPTON, KANDOR, and CLASSIC, limit expressive power of the knowledge representation language in an effort to achieve tractable subsumption algorithms [59]. However, this approach results in languages so limited that they are no longer generally useful. KRYPTON, one of the few languages (if not the only language) to achieve a tractable subsumption algorithm, never found its way into applications, partially because of the limited expressiveness of its knowledge representation language [59].

A second limitation of traditional classifiers is that they use ill-characterized subsumption algorithms. These systems use the following criterion for subsumption [7]:

> A concept X subsumes a concept Y if and only if, in all possible interpretations, the extension of X is a superset of the extension of Y.

Woods calls this definition of subsumption *extensional subsumption* [58]. This criterion for subsumption leads to undecidability for languages similar to ours, such as NIKL [44] and KL-ONE [48] (see [2] for details of our knowledge representation language). Even for languages with limited expressiveness, such as many of the descendents of KL-ONE, extensional subsumption leads to intractability [59, 38]. As a result, most classifiers have retreated to tractable but incomplete subsumption algorithms [43]. These algorithms are sound with respect to the above subsumption criterion, but they lack a precise specification of what subsumption relationships they detect (their degree of completeness). This is surprising given the strong KL-ONE tradition of grounding the representational system in formal semantics. (A notable exception is Patel-Schneider's approximate account of the subsumptions that the NIKL classifier detects [44].)

The third limitation of traditional classifiers is that they are based on the extensional subsumption criterion, but they are restricted to using only terminological (*i.e.,* definitional) knowledge, knowledge that

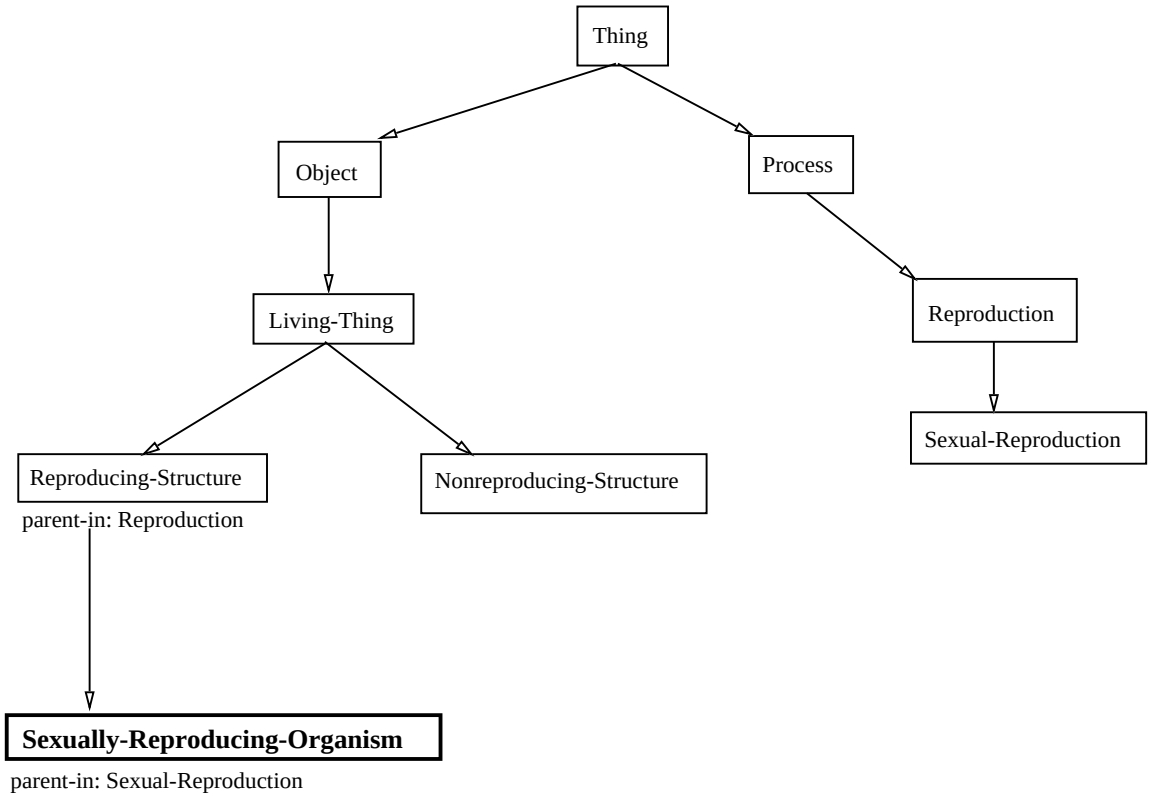


Figure 2: Knowledge base fragment used to illustrate content addressability for *Sexually-Reproducing-Organism*, described by (Living-Thing (parent-in Sexual-Reproduction)). Only the relevant features of each concept are shown.

as necessary or sufficient on that frame is present in the description and every feature in the description is present on the frame and is both necessary and sufficient. For example, for the *Living-Thing* frame to match the above description, *parent-in* =*Sexual-Reproduction* must be necessary and sufficient for *Living-Thing* and *Living-Thing* must have no other defining features. In this example, the match fails, so KASTL proceeds to the next step.

Step 4 of the procedure is to examine each immediate specialization of the base concept for a match. If a match is found, it is returned. Otherwise, KASTL repeats steps 3 and 4 for each specialization whose definition is more general than the given concept description. A specialization S is more general than the concept description if S has at least one sufficient feature (*i.e.,* S has a definition) and each such feature appears in the concept description or generalizes some feature in the concept description. If no specialization of the base concept is more general than the given description, KASTL fails to find an exact match.

The knowledge-base fragment shown in Figure 2 can be used to illustrate step 4 for the concept description (Living-Thing (parent-in Sexual-Reproduction)). The search for a match is restricted to the portion of the taxonomy rooted at *Living-Thing*. KASTL first attempts to match each specialization of *Living-Thing* with the given description. Neither *Nonreproducing-Structure* nor *Reproducing-Structure* is an exact match, but *Reproducing-Structure* is more general than the concept description, because the feature *parent-in* = *Reproduction* on *Reproducing-Structure* subsumes the feature *parent-in* = *Sexual-Reproduction* in the concept description. Therefore KASTL repeats the matching process for the specializations of *Reproducing-Structure*. On this iteration a match is found, *Sexually-Reproducing-Organism*.

If KASTL fails to find an exact match for a given concept description, the described concept is not explicitly represented in the knowledge base. KASTL's response to this failure depends on whether the user asked for an exact match or was willing to settle for a partial match. If asked for a partial match, KASTL returns the list of (maximally general) concepts in the knowledge base that are more specific than the described concept. This facility allows users to locate a concept through a description more general than the one that uniquely identifies a concept, and it allows users to access by description concepts that cannot be completely defined (e.g. "natural kinds"). KASTL's algorithm for searching the knowledge base for partial matches is a slight variant of step 3 of KASTL's algorithm for accessing the virtual knowledge base (shown in Figure 3 and discussed in the next section) in which the search originates with the base concept.

If, on the other hand, the user asked KASTL to find the frame that matches the concept description exactly, KASTL accesses the concept in the virtual knowledge base, as described next.

2.2 Accessing Concepts in the Virtual Knowledge Base

The task of accessing concepts in the virtual knowledge base can be described informally as "given a description of a concept, modify the knowledge base to include that concept." A more precise formulation of the task as performed by KASTL is shown below:

Given: A concept description consisting of

- A base concept, and
- A set of features (*i.e.,* slot-value pairs),

Return:

1. Insure that the given concept description is meaningful. Each frame name and slot name must exist, and each feature must be valid for the specified base concept.
2. Convert the base concept B to a more specific concept, B', if possible, by examining the features given in the description and the constraints contained in the knowledge base.
3. If B' matches the input description, return B'.
4. If B' is more general than the input description, then repeat steps 3 and 4 for all specializations of B'.

Figure 1: KASTL's procedure for providing content addressability.

(Water (transportee-in (Diffusion (source Soil-Region) (destination Plant))

describes "water transported by diffusion from the soil into a plant." KASTL matches nested descriptions from the inside out. For example, KASTL first searches for a frame matching

(Diffusion (source Soil-Region) (destination Plant)),

such as *Plant-Water-Uptake*. It then substitutes that frame name for the nested description to yield

(Water (transportee-in Plant-Water-Uptake)).

Finally, KASTL searches for a frame matching the modified description.

In addition to concepts, users can also access *slots* in the virtual knowledge base by their description. A list of slots in place of a slot name refers to the disjunction of those slots. For example, (*husband wife*) would refer to the slot *spouse*. Users can also refer to the transitive closure (Kleene star) of a slot. For example, (transitive-closure-of *parent*) would refer to the slot *ancestor*.

Figure 1 gives the procedure KASTL uses to provide content addressability for exact matches. The first step is to insure that the given concept description is meaningful. This involves checking that each frame name and slot name in the description exists and that each feature is valid. A feature *slot* = *value* is valid when *value* is in the *range* of *slot*. For example, the description (Glucose (product-of Photosynthetic-Cell)) is not valid if slot *product-of* has range *Process*, because *Photosynthetic-Cell* is not a *Process*.

The next step is an efficiency measure. To reduce search, KASTL converts the base concept within the given concept description to a more specific concept if the conversion does not change the meaning of the description. For example, given the concept description (Object (parent-in Sexual-Reproduction), "an object that reproduces sexually," KASTL modifies the description to (Living-Thing (parent-in Sexual-Reproduction)). Changing the base concept from *Object* to *Living-Thing* does not change the meaning of the description because KASTL determines from examining the knowledge base that only a living thing can reproduce (*i.e.,* the *domain* of slot *parent-in* is *Living-Thing*). This modification greatly simplifies the search for a match; specializations of *Object* that are not specializations of *Living-Thing* need not be examined.

The third step of the procedure is to determine whether the (possibly modified) base concept matches the given description. For example, KASTL determines whether the *Living-Thing* frame matches the description (Living-Thing (parent-in Sexual-Reproduction)). A frame matches the description if every feature annotated

knowledge base or the virtual knowledge base). For example, if the concepts *Eukaryotic-Cell* and *Cytoplasm* and the relation *part-of* are reified in the knowledge base, then the concept "Cytoplasm of a Eukaryotic Cell" can be completely defined. Hence, it can be uniquely identified by description (if it is already reified) or created (if it is not). Furthermore, since this concept is in the virtual knowledge base, it can be used to define other concepts that are also in the virtual knowledge base, and so on.

Although only concepts that can be completely defined can be uniquely identified in a content addressable, virtual knowledge base, such concepts appear to occur frequently enough to make developing methods for accessing them worthwhile. In an analysis of a chapter from a biology textbook, of the 899 concepts referenced in 55 paragraphs, approximately 29% of them referred to concepts that could be completely defined.

The second mode of operation for our access methods is to find the concept(s) in the knowledge base that *best* match the input description. This mode of operation is useful for finding concepts that have no strict definitions, such as "natural kinds." The Krypton system also distinguished between concepts with complete definitions (which were represented in Krypton's TBox) and those with partial definitions (which were represented in the ABox) [6].

2.1 Accessing Concepts by Description (Content Addressability)

The task of accessing concepts by description can be described informally as "given a description of a concept, find the knowledge-base frame that represents the concept." A more precise formulation of the task as performed by KASTL is given below:

Given: A concept description, in a formal language, consisting of

- a base concept B (more precisely, the name of the frame representing B),
- a set F of features (*i.e.,* slot-value pairs), and
- *mode*: exact match or best match

Return: The name of the frame representing the concept C that matches the description. C matches the description if and only if

- C is a specialization of B (although not necessarily an immediate specialization), and
- all features in F are necessary for membership in C and (if *mode* = "exact match") the features in F are jointly sufficient for membership in C for all members of B.

For example, in the concept description (Cell (part-of Plant)), the base concept is *Cell* and the only feature is *part-of* = *Plant*. The task is to find a frame that represents a specialization of *Cell* for which the feature *part-of* = *Plant* is necessary and (for exact match) sufficient. In other words, the task is to find the frame representing the concept whose definition is "cell that is part of a plant."

Concept descriptions may have multiple features, as in

(Cell (part-of Plant) (producer-in Photosynthesis)).

Multiple features are interpreted conjunctively; *all* of the specified features must appear on the matching concept. Thus, the above example describes "photosynthetic plant cell." Concept descriptions may also be nested. For example, the description

2 A Content Addressable, Virtual Knowledge Base

While representing knowledge, a knowledge engineer makes numerous decisions, many of them arbitrarily. For example, the choice of what name to give each frame is often arbitrary (*e.g.*, "Plant-Stem" vs. "Stem-of-Plant"). Similarly, the choice of which domain concepts to reify (create a frame for) in the knowledge base depends on the knowledge engineer's subjective judgment of the relative importance of concepts. For instance, the knowledge engineer might create a frame for *Condensation* and a frame for *Water*, but not a frame for *Water-Condensation*. Because relative importance varies from one task to another, decisions the knowledge engineer makes regarding which concepts to reify in a knowledge base will not be appropriate for all tasks in all situations. A goal of this research is to insulate users of the knowledge base from the effects of the (sometimes arbitrary) choices made during knowledge representation.

To insulate users from the internals of the knowledge base, our access methods provide an abstraction of the knowledge base, a *content addressable, virtual knowledge base.* A content addressable knowledge base allows users to locate a frame using a partial description of the frame's contents. For example, to locate the frame for "plant cell," the user might provide the description "cell that is part of a plant." When given this description in place of a frame name in an access request, the access method searches the knowledge base for the frame that matches the description. It then uses the name of that frame in servicing the request.

A virtual knowledge base is one that contains all the concepts and facts that are implicit in the actual knowledge base. In the actual knowledge base, the only concepts that are accessible are those that are reified as frames, and the only facts that are accessible are those represented by an explicit $\langle frame, slot, value \rangle$ triple. In a virtual knowledge base, by contrast, concepts and facts that are implicit in the knowledge base are also accessible. That is, the virtual knowledge base consists of all concepts that can be defined in terms of other concepts and relations in the knowledge base, and all facts implied by other facts in the knowledge base. Several methods exist for accessing *facts* in the virtual knowledge base (inheritance, rule chaining, etc.). This research provides methods for accessing *concepts* in the virtual knowledge base.

When an access method provides both content addressability and access to concepts in the virtual knowledge base, users need not know whether concepts are explicit in the knowledge base or where they are located. Users simply supply a concept description embedded in an access request. If the concept has a frame associated with it, then the system will find and use that frame to service the request. Otherwise, the system will create and use a new frame. From the user's point of view, there is no distinction between accessing concepts by description and accessing concepts in the virtual knowledge base.

For an access method to provide content addressability and to provide access to concepts in the virtual knowledge base, it must determine, for each concept in the knowledge base, whether the input description matches it exactly, is more general than it, is more specific than it, or is neither more general nor more specific. This step is called *subsumption* [59, 58]. The subsumption calculation is deterministic only when the input description provides a complete definition of the concept to be accessed, one containing necessary and sufficient criteria that completely delineate the concept. (It is not possible to determine whether two *partial* definitions describe exactly the same concept.) This implies that, although any concept in the knowledge base can be identified as a *potential* match for a given description, only completely defined concepts can be *uniquely* identified by description (*i.e.,* identified as the only possible match for the description), and only completely defined concepts exist in the virtual knowledge base.

For this reason, our access methods operate in two modes. First, they can find or create the concept that *exactly* matches the input description. This mode of operation requires that the input description completely define the desired concept in terms of other concepts and relations in the knowledge base (either the actual

simulation [47], machine learning [37], and supporting knowledge engineers [30]), we have found that current access methods for structured knowledge bases are inadequate. The goal of this research is to improve them.

Consider, for example, locating information about "Leaf photosynthesis" from a knowledge base of frames using conventional methods. The first problem the user faces is that conventional access methods require the user to provide the address within the knowledge base of the data object (henceforth, "frame") that represents the concept. This address might be the frame's name, such as *Leaf-Photosynthesis*, *Photosynthesis-In-A-Leaf*, or *Leaf-Energy-Production*, or it might be a path consisting of frames and slots, such as *(Leaf producer-in).* Because the names of frames and slots are often arbitrary, it is problematic for a user to locate a frame by its address.

The second problem faced by the user attempting to locate information about "Leaf photosynthesis" is that a frame representing the concept might not exist in the knowledge base. Knowledge engineers decide which concepts to explicitly represent (reify) based on *a priori* assessments of importance, which are often inappropriate for particular users or tasks. Consequently, the knowledge base might contain frames representing the concepts "Leaf" and "Photosynthesis," but not "Leaf photosynthesis." With traditional access methods, knowledge base users can only overcome these two problems by increasing their familiarity with (and thus, their dependence on) the internal details of the knowledge base.

In contrast, our solution to these problems in locating information is to provide, through our access methods, an abstraction of the knowledge base. Our methods, implemented in a program named KASTL, locate frames by their contents (in addition to locating frames by their addresses, as with conventional methods). The user provides a partial description of the desired concept, such as "photosynthesis in which the producer is a leaf," and KASTL searches the knowledge base for the frame representing that concept. If no such frame exists in the knowledge base, KASTL creates one and splices it into the taxonomy to yield a *virtual knowledge base.*

After locating a concept, our access methods provide an additional service: extracting *viewpoints*, coherent sets of facts that describe a concept from a particular perspective. We have identified many types of viewpoints and developed methods for extracting them from structured knowledge bases, either singly or in combinations. For example, our methods can extract many viewpoints of *leaf*, such as a leaf's structure and its functional role in transpiration. In contrast, conventional access methods either extract a single fact (when asked for the the filler of a particular frame-slot) or they extract all the facts about a concept (when asked for the frame representing the concept). Our evaluation indicates that viewpoints extracted by KASTL are comparable in coherence to those people construct.

These capabilities — locating concepts by description and extracting viewpoints of them — are particularly important for users accessing large knowledge bases. A large knowledge base is costly to search, prohibitively so if the knowledge base lacks structure (*i.e.,* if it is represented with first order logic or some other nonindexed formalism). For this reason, our research assumes a structured knowledge base. A structured knowledge base is inherently easier to search because the taxonomy of frames provides an index: each frame collects all the information about the concept it represents. This research addresses the problem of automatically extending this index when the user accesses an implicit concept, and it addresses the problem of selecting coherent portions of the extensive information about each concept.

Access Methods for Large Knowledge Bases*

Liane Acker
IBM Corporation
11400 Burnet Road
Austin, Texas 78758
acker@austin.ibm.com

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

February 3, 1993

Abstract

The access methods for a knowledge base provide ways to locate information. Although they are critically important for virtually all applications of a knowledge base, current methods are inadequate, especially for large, structured knowledge bases. To locate information about a concept using current access methods, the user must provide the address (usually a frame name) of the concept within the knowledge base, which requires an unrealistic level of omniscience. Moreover, only those concepts reified in the knowledge base can be located, which excludes information about the many concepts that are implicit in the knowledge base. Our solution to these problems is to provide, through our access methods, an abstraction of the knowledge base, one in which concepts can be located by a partial description of their contents and in which implicit concepts are automatically reified when they are requested.

After locating a concept, our access methods provide an additional service: selecting coherent subsets of facts about the concept. Conventional methods either return all the facts about the concept or select a single fact (usually the filler of a specified frame-slot). Our access methods extract *viewpoints* — coherent collections of facts that describe a concept from a particular perspective. We have identified many types of viewpoints and developed methods for extracting them from knowledge bases, either singly or in combinations. Our evaluation indicates that viewpoints extracted by our methods are comparable in coherence to those people construct.

1 Introduction

The access methods for a knowledge base provide ways for users (both knowledge engineers and application programs) to locate information. While building a large knowledge base for one area of biology [45] and developing systems to perform a variety of tasks using that knowledge base (such as tutoring [1, 50, 26],

*Support for this research was provided by an IBM Graduate Fellowship to Liane Acker, a grant from the National Science Foundation (IRI-8620052), a contract from the Air Force Human Resources Laboratory (F33615-90-C-0014), and donations from Digital Equipment Corporation and the Lockheed AI Laboratory. This research was conducted in the Department of Computer Sciences, University of Texas at Austin.