Using Analogies in Natural Language Generation

Vibhu O. Mittal and **Cécile L. Paris**
USC/Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292
U.S.A.

Abstract

Any system with explanatory capabilities must be able to generate descriptions of concepts defined in its knowledge base. The use of analogies to highlight selected features in these descriptions can greatly enhance their effectiveness, as analogies are a powerful and compact means of communicating ideas and descriptions. In this paper, we describe a system that can make use of analogies in generating descriptions. We outline the differences between using analogies in problem solving and using them in language generation, and show how the discourse structure kept by our generation system provides knowledge that aids finding an acceptable analogy to express.

Introduction

Good explanation capabilities are crucial for many systems, including expert systems, intelligent tutoring systems and other Knowledge-Based Systems (e.g., to provide on-line documentation). Many such systems can already communicate concepts or processes coherently and effectively to their users. Communication would be enhanced further if a system could employ analogies to help describe concepts and highlight its features. This paper describes a system that can employ analogies in generating descriptions.

Analogies have long been recognized as a valuable tool in problem solving, and it is mainly in this perspective that they have been studied in artificial intelligence – e.g., (Carbonell, 1986; Prieditis, 1988; Falkenhainer *et al.*, 1989); their use in explanation has largely been overlooked. In this paper, we address the problem of finding and using appropriate analogies efficiently to describe concepts. Finding an analogy is computationally expensive: the process essentially involves an attempt to find a consistent mapping between two concepts and their associated relations, sometimes with only a partial and fragmented knowledge about one concept. It is especially expensive if an *exact* mapping is desired, as in the case of most analogical problem solving (APS) systems. However, if *approximate* matches are deemed acceptable, the cost of finding an analogy can be substantially reduced. We illustrate with an example that a generation system does not in fact require the sort of exact matches that most APS systems require.

Our framework is designed around an already existing explanation facility (Moore & Paris, 1989; Moore & Swartout, 1991; Paris, 1991), which is part of the Explainable Expert System (EES) framework (Neches *et al.*, 1985; Swartout *et al.*, 1991). We show how the knowledge about the utterance to be generated and its hierarchical structuring in the form of rhetorical goals can help us identifying and using analogies in the context of explanation generation.

Analogies in Problem Solving vs in Text

Several major differences make the use of analogies in discourse much easier than in problem solving:

First, the features to use to find the analogy are more clearly determined. Unless an APS system builds an explanation tree, there is *a priori* little or no indication of which features are relevant in different situations. Yet, because of the computational cost (since finding an analogy requires searching through a large space of concepts and relations, the fewer the constraints to satisfy, the faster analogies can be found), a minimum number of features should be used in the matching process. Domain-independent theories of feature relevance have been proposed to address this problem: the use of higher-order relations in Gentner's Structure Mapping Engine (SME) (Gentner, 1983; Falkenhainer *et al.*, 1989); the explanation-proof trees by Kedar-Cabelli (1988); the use of abstractions (Greiner, 1988) and justifications of the problem solving (Carbonell, 1986). In discourse generation however, the most important features to match on are given: they are the ones the system is trying to illustrate. Given an appropriate representation of the explanation being constructed, the system can determine the features to highlight or elaborate upon, and use these to find an analogy. Thus, a system designed to employ analogies in generation does not face one of the more difficult problems that APS systems have to address – that of determining the important features on which to match.

†The authors can be contacted through electronic mail at: {MITTAL,PARIS}@ISI.EDU. Vibhu Mittal is also in the Computer Science Department of the University of Southern California.

Second, should a system not find a suitable analogy for all the given features, it *should* be able to use analogies for subsets of the original features. Since the determination of these subsets is quite domain specific, most APS systems do not possess a theory of partitioning the features. However, an appropriate representation of the text being constructed contains sufficient information to determine at least one possible subset partitioning for renewed matching attempts in most cases, and thus analogies for a subset of the original features can be used.

Finally and most importantly, expository systems can make use of partial matches, because people seem to have little difficulty in understanding complex and incomplete mappings, as in: *"He is big, and burly, like a bear"*, without trying to resolve how this mapping would affect attributes not mentioned in the sentence. This is *not* possible in APS systems. In addition, if there is a point that is likely to be mis-understood in the analogical mapping, the system can generate an explicit concessionary clause for that point (*"Though A is like B, A is not a ... "*). A system that needs to find analogies for communication purposes therefore has far greater leeway in finding potential matches than an APS system which needs analogues that are consistent in their mapping across many more features.

In summary, then, expository systems can have more guidance as to how to find an analogy and more flexibility in their choice and application of analogies than most APS systems. In the following sections, we briefly describe our generation framework and illustrate our method with simple examples.

The System

Our system is built on an extensive framework for generating explanations for expert systems in natural language using the EES expert system environment (Neches *et al.*, 1985; Moore, 1989; Moore & Swartout, 1991; Paris, 1991; Swartout *et al.*, 1991), the PENMAN Natural Language Generation system (Mann, 1983), and the LOOM knowledge representation language (MacGregor, 1988). A system built using EES can generate descriptions of the terminology it uses, because it has a representation for all its concepts in a terminological knowledge base. The quality of descriptions generated thus depends upon both the detail and the accuracy of the domain knowledge representation. The representation of the terminological knowledge in EES is influenced not only by its operationality in problem solving, but also by the necessity of generating good explanations in the form of descriptions about the concepts themselves. Thus, one of the essential requirements for any system using analogies, a well-detailed domain model, is provided.

To generate grammatically correct English text, the domain model is anchored beneath the Upper-Model, a computational resource for organizing domain knowledge appropriately for linguistic realizations (Bateman *et al.*, 1989). The Upper-Model is a knowledge base of general conceptual categories that provides a domain- and task-independent classification system which supports sophisticated natural language processing.

Given a top level intentional (or discourse) goal to be achieved, text is generated by hierarchical goal decomposition. To ensure the coherence of the text, the text planner makes sure that two subgoal siblings are related with a rhetorical relation, which indicates the functional relationship between any two text spans. Such relations include, for example, `contrast`, `motivation` and `elaboration`. The rhetorical relations used in our system are based on Rhetorical Structure Theory, a theory of coherence (Mann & Thomspson, 1988). As a result of the text planning process, a detailed text plan or Discourse Structure Tree is produced. This text plan contains all the intentional goals and subgoals, as well the rhetorical relations between subgoals. Subgoals in the plan are annotated as either a NUCLEUS or a SATELLITE, based on their semantic relationship with their parent goal. Nucleic subgoals represent the core or the focus of the information to be presented; satellite subgoals play a supporting role to the nucleus, their relationship being indicated by a rhetorical relation.[1]

Using the Discourse Structure to Find Analogies

There are two ways in which the discourse structure tree (DST) generated by the text planning process can be used to find analogies (which are then incorporated into the DST): (1) to find the features on which to match (2) to find relevant sub-groupings of features to find partial analogies. More specifically, the rhetorical information (RST relations like `elaborate` and `contrast`) and the subgoal specification (NUCLEUS vs SATELLITE) contained in the DST tree can be used to focus the search for analogies. This section describes the algorithm which makes use of these features.

Our framework uses a slightly modified version of SME to actually find the analogies. Among the differences: during matching, only the features posted are used, and not all the ones that the concept possesses; furthermore, the Upper-Model, as a set of domain independent abstractions that link different knowledge bases together, is used as an additional set of constraints (this enables the system to find analogical concepts that are realized in similar ways in language). Since the system is capable of handling imperfect matches, an efficient but non-optimal algorithm – such as the greedy version of SME (Forbus & Oblinger, 1990) – can be used. It is important to note that the knowledge bases in EES are organized hierarchically, subordinated under the Upper-Model. Although the Upper-Model is by itself too general to work as an effective abstraction hierarchy (as used by Greiner, for instance), it is very useful in categorizing concepts, relations and properties into classes, which can be used to restrict the search. The actual algorithm is presented in Figure 1.

Since the number of features used to constrain the match is relatively small as compared to the number of features employed in APS systems, it is possible to find a relatively

[1] A detailed description on the text planner is out of the scope of this paper. See (Moore, 1989; Moore & Paris, 1989; Moore & Swartout, 1991; Paris, 1991).

Algorithm:

1. Take the text-plan for the text to be generated and scan it to see if there is a concept which is flagged. Concepts are flagged if they are one of the following:
   (a) Concepts for which analogies have been used in the past dialogue.[3]
   (b) Concepts whose parents have been described using analogies.
   (c) Concepts which are *parts* of components that have been described using analogies.
   (d) Concepts for which there are no lexical-items.[4]

   For any flagged concepts, retrieve the generated analogy and see if it can be extended for this case too. If it cannot, see if a related analogy (that is an analogy from the same domain) can be used in the current text-plan.

2. Traverse the text-plan in a depth-first manner. If there are any CONTRAST or ELABORATION rhetorical relations between two sub-goals posted, attempt to find analogies as follows:
   - **ELABORATION:** Consider the ELABORATION relation and the two sub-goals it relates, as shown in Figure 5, Part (a). The sub-goal on the left-hand side (goal-1) presents some information about a concept, which the sub-goal on the right-hand side (goal-2) elaborates upon. Goal-1 is referred to as the NUCLEUS, and Goal-2 is referred to as the SATELLITE. To generate analogies, the system takes the NUCLEUS sub-tree and gathers all the features that it is supposed to communicate. It also collects all the features expressed by the SATELLITE (to elaborate on the NUCLEUS). The NUCLEUS features are used to find an analogy (necessary features). The SATELLITE features are used as *constraints* that can be relaxed. If an analogy *is* found, it is incorporated (as described below) into the text-plan.
   - **CONTRAST:** The contrast relationship between the two sub-goals indicates that the features expressed by the two sub-goals are contrasting ones. In this case, one can generate analogies for either or both of the sub-goals. However, because of the contrasting nature of the features expressed in the two sub-goals special care must be taken in this case. The two sub-goals are inter-dependent, and an analogy for one of the sub-goals must take into account this dependency. There are two ways in which an analogy may be generated here:
     (a) An analogy can be found by using the features mentioned in one of the sub-goals *and* the negation of the features mentioned in the other sub-goal.
     (b) Two analogies can be found, one for each sub-goal. Using such a compound analogy is allowed only if the two analogies generated belong to the same domain. An example of such a compound analogy is: "Even though he was as big and tough as a *bear*, he was as tender and gentle as a *lamb*". In this case, the analogues bear and lamb are both from the animal domain.

3. For each analogy proposed, evaluate it as follows: If all the features to be presented in the NUCLEUS and the SATELLITE clauses map consistently, incorporate the analogy in the text-plan without further modifications. If, on the other hand, there are any mentioned features that do not map over, generate a CONCESSIVE clause to be appended to the clause containing the analogy ("but ... ," "except ... ," "although ... ," etc).

Figure 1: Algorithm for incorporating analogies with CONTRAST and ELABORATION relations.

large number of analogues. In such a case, there are two alternatives:

1. The system first checks to see whether other analogues are marked POSSIBLE for some other node in the discourse tree. If the current node is related (either a sibling or an ancestor), the system prunes the set of analogues possible for the current node by keeping only those analogues that are compatible with those selected for the related node.
2. If there are no other analogues, the system marks the current set as POSSIBLE, and arranges the analogues such that the number of features that match with the base-concept are maximized, while at the same time minimizing the number of dis-similar features.

In case no suitable analogues are found, the system attempts to determine subgroups of features to use for matching partial analogies.

To clarify these two issues (splitting features and selecting from multiple analogues), consider the following example from the domain of local area networks (DEC Manual,1987).

... A communication line can be of three types: simplex, half-duplex and full-duplex ... A simplex line supports data flow in one direction only ... a half-duplex line supports data flow in two directions, but transmissions can only occur in one direction at a time ... a full-duplex line supports data flow in both directions at the same time ...

[3]Our system retains text plans in order to participate in a dialogue, as described in (Moore & Paris, 1989; Moore, 1989).

[4]Our system cannot currently generate these types of analogies.

It is possible that an attempt to find a suitable analogy for the `communication-line` fails. However, the system can then attempt to determine whether partial analogies can be used to describe the concept. From the DST, the system determines that a communication line is to be described as having three sub-types: simplex, half-duplex or full-duplex. This raises the possibility of finding an analogy for *part* of the original concept. The system tries to find analogies for each of these types, using the features that are most important for each type. The search returns a number of possible analogues for the simplex line: {`electronic-diode, valve, one-way-street, ...`}. This set is marked as POSSIBLE, and the system attempts to find analogies for the half-duplex and full-duplex lines. It finds the sets {`two-way-street-with-one-lane-obstructed, ...`} and {`hosepipe, electric-wire, two-way-street`} respectively. The algorithm constrains the system to pick analogies such that related nodes are expressed by related analogies if possible. Since the three nodes are siblings in this case, the system picks a set of related analogies (in this case, sibling concepts subsumed by the concept of `street`). The DST is then modified by the insertion of plan fragments representing these three analogies in the original description. The description generated would be similar to the following, the original description taken from the DEC manual (1987) (p. 2-12):

... A communication line can be of three types: simplex, half-duplex and full-duplex. A simplex line supports data flow in one direction only. This type of signal flow can be compared

to traffic down a one-way street. A half-duplex line supports data flow in two directions, but transmissions can only occur in one direction at a time. This type of signal flow can be equated to traffic on a two way street with an obstruction in one of the lanes. Two way traffic is possible, but drivers headed in opposite directions must alternate since they have only one lane to use. A full-duplex line supports data flow in both directions at the same time. This is similar to traffic on a two-way street with no obstructions.

As this example demonstrates, the finding of suitable analogies in an explanation framework which maintains a suitable discourse structure can be done quite flexibly, because of the possibility of splitting up the concept and using fewer features (and constraints) with which to find analogies. The following section deals with a short example that illustrates how the algorithm can also make use of the information in the rhetorical relations between clauses to focus the search for analogies.

Focusing the Search with RST Relations

The previous section illustrated how the algorithm could be used to find analogies by splitting up the number of features required to match. There is yet an additional source of information in the discourse structure that can be used: the rhetorical relations that hold between the NUCLEUS and SATELLITE nodes at various levels of the DST. We now illustrate this point with an example, which, for the sake of clarity, only uses a small hypothetical text plan.

Suppose an expert system is asked to select a person for a particular task. The terminological knowledge base of the system will include descriptions of `man`, `person`, etc. The specific domain model will include specific facts about individuals. A portion of the knowledge base together with some specific facts about Mr. Smith is shown in Figure 2. In this case, Mr. Smith is shown as being strong, burly, intelligent, etc. Concepts such as `person` are defined in the Upper-Model (as indicated by the `UM-KB:` prefix in the symbol name). Other domain specific concepts and roles are defined in terms of the Upper-Model to facilitate the generation of appropriate English.

Suppose the system needed to communicate that the assignment needed two features – strength and tactfulness, and that Mr. Smith had these features. It could generate the text shown in Figure 3, Part (a). To generate this text, the system would choose the ELABORATION relation to express these properties of Mr. Smith, as shown in the fragment of the text plan in Figure 5, Part (a).

To enhance this text with an analogy, the system can try to find a concept that matches all the features to be communicated. Failing to do so, the system attempts to find concepts that match partial sets of features, as determined by the features expressed in the NUCLEUS and the SATELLITE, starting with the NUCLEUS, as these are the main features to illustrate, the features of the SATELLITE being considered as desirable but not mandatory. It does find an analogy (the bear), but realizing that the `softspoken` feature is not present in the analogue, generates a CONCESSIVE clause and

```
(defconcept PERSON
  :is (:and UM-KB:Conscious-Being)
  :disjoint-covering
      (UM-KB:Female UM-KB:Male))

(defconcept man
  :is (:and :primitive UM-KB:Male))
:
(tellm (man Smith))
(tellm (:about Smith
               (:characteristics
                strong burly
                humorous tactful
                gentle softspoken)
               (:UM-KB:age 42)
               (:UM-KB:size big)
               (:profession scientist)
               (:likes caviar)

...))
```

Figure 2: A fragment of the knowledge base in LOOM notation.

inserts it into the text (instead of the ELABORATION). The text generated is shown in Part (b) of Figure 3.

However, based on the context, the system might try to highlight the tactful quality of Mr. Smith. Figure 4, Part (c), shows what text the system generates when it plans the same text using a CONTRAST rhetorical relation so as to highlight the `softspoken` feature. Finally, Figure 4, Part (d), shows the text generated when the system decides to use analogies on both sides of the rhetorical relation. It uses big, strong and not-tactful as features to map on the left-hand side (the NUCLEUS) and softspoken, *not*-big and *not*-strong as the features to match on the right-hand side (the SATELLITE). The system finds many analogies for the left right parts (for example "big and strong as a *rock* ..."), but since the soft-spoken property maps to fewer targets, and the one used, a lamb, is an animal, the other analogue picked is the bear. Figure 5, Part (d), shows the relevant fragments of the skeletal text plan generated in this case.

This point illustrates one of the differences between the way this system uses analogies and the way they are used in APS systems. Here, for instance, the system attempts to find an analogy at the level of the CONTRAST relation. Not finding another concept whose features match the desired characteristics, the system breaks up the features to be matched into smaller subsets (big and strong, soft-spoken or tactful) to see whether an analogy can be found for any of these subsets of features. This is not usually possible in APS systems, since they usually need an analogy which can map *all* the necessary features, and partial matches cannot be used (because of their un-matched attributes).

System: Selected for assignment: John Smith
User: Why?
System: Because the assignment requires both strength and tactfulness. John is both strong and softspoken.

Part (a): No emphasis on `softspoken`. The text is planned using the ELABORATION relation.

User: Why?
System: John Smith has been selected because the assignment requires strength and tactfulness. He is big and strong *like a bear*, but he is softspoken.

Part (b): Using an analogy – the `big` and `strong` features matched. A CONCESSIVE clause was generated for the `softspoken` feature.

Figure 3: Generating an analogy in the case of an ELABORATION relation.

User: Why?
System: John Smith has been selected because the assignment requires strength and tactfulness. Even though he is big and strong, he is very softspoken.

Part (c): Using the CONTRAST rhetorical relation to highlight `softspoken`.

User: Why?
System: John Smith has been selected because the assignment requires strength and tactfulness. Even though he is big and strong *like a bear*, he is as softspoken *as a lamb*.

Part (d): Generating analogies for both clauses related by a CONTRAST relation, to highlight the `softspoken` feature.

Figure 4: Analogy in the case of an CONTRAST relation

Conclusions

Generating effective, understandable descriptions of concepts and terms in the knowledge base is an important requirement for tasks such as explanation, intelligent tutoring and automatic documentation. In this paper, we have described how, by keeping an appropriate representation of the discourse, a generation system can make use of analogies without incurring all of the overhead that APS systems normally do.

The major difference between generating expository analogies for explanation and finding analogies for problem solving lies in two facts: (1) that expository analogies need to be consistent only about a very small region of the feature space and still be understood, and (2) an appropriate discourse representation provides information to partition the set of features to search for partial analogies and to determine which features are mandatory as opposed to desirable in the search for the analogy (features in the NUCLEUS as opposed to features in the SATELLITE). In generation, then, a system has more information to allow it to find analogies and more flexibility in its use of analogies. Our use of analogies in generating discourse also demonstrates the importance of having task specific criteria for finding and using analogies efficiently and effectively.

We have shown how the relevant features may be extracted from the information available to the text planner. We have also shown that the locality of the information in discourse together with the rhetorical structure can provide information as to the importance of the features to be used in the match process and can thus help focus the search. While there are some unresolved questions about whether the analogies should be generated and integrated with the rest of the text while constructing the text plan, or as our system currently does, by super-imposing the analogy in the text and modifying the text plan, our system demonstrates that it is possible to use analogies in generating discourse in a practical and effective manner.

Acknowledgments

This work was supported in part by the NASA-Ames grant NCC 2-520 and under DARPA contract DABT63-91-C-0025. We are grateful to all members of the EES Project, past and present, for much of the framework's foundation. Our sincere thanks to Profs. Forbus and Gentner for providing us with a version of SME.

References

(Bateman *et al.*, 1989) Bateman, J., Kasper, B., Moore, J., & Whitney, R. (1989). *A general organization of knowledge for natural language processing: The Penman Upper Model.* USC/Information Sciences Institute.

(Carbonell, 1986) Carbonell, J. G. (1986). Derivational Analogy: A Theory of Reconstructive Problem Solving and Expertise Acquisition. In R. Michalski, J. Carbonell, & T. Mitchell (Eds.), *Machine Learning*, volume II chapter 14. Los Altos, CA.: Morgan Kaufmann Publishers.

(DEC, 1987) Network Training Solutions: Introduction to Data Communications – Student Guide. Digital Educational Services, Serial No. EY-6716E-SG-0001.

(Falkenhainer *et al.*, 1989) Falkenhainer, B., Forbus, K. D., & Gentner, D. (1989). The Structure Mapping Engine: Algorithm and Examples. *Artificial Intelligence, 41*(1), 1–63.

(Forbus & Oblinger, 1990) Forbus, K. D. & Oblinger, D. (1990). Making SME Greedy and Pragmatic. In *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, (pp. 61–69)., Boston, MA. Lawrence Erlbaum Associates, Publishers.

(Gentner, 1983) Gentner, D. (1983). Structure Mapping: A Theoretical Framework for Analogy. *Cognitive Science, 7*(2), 155 – 170.

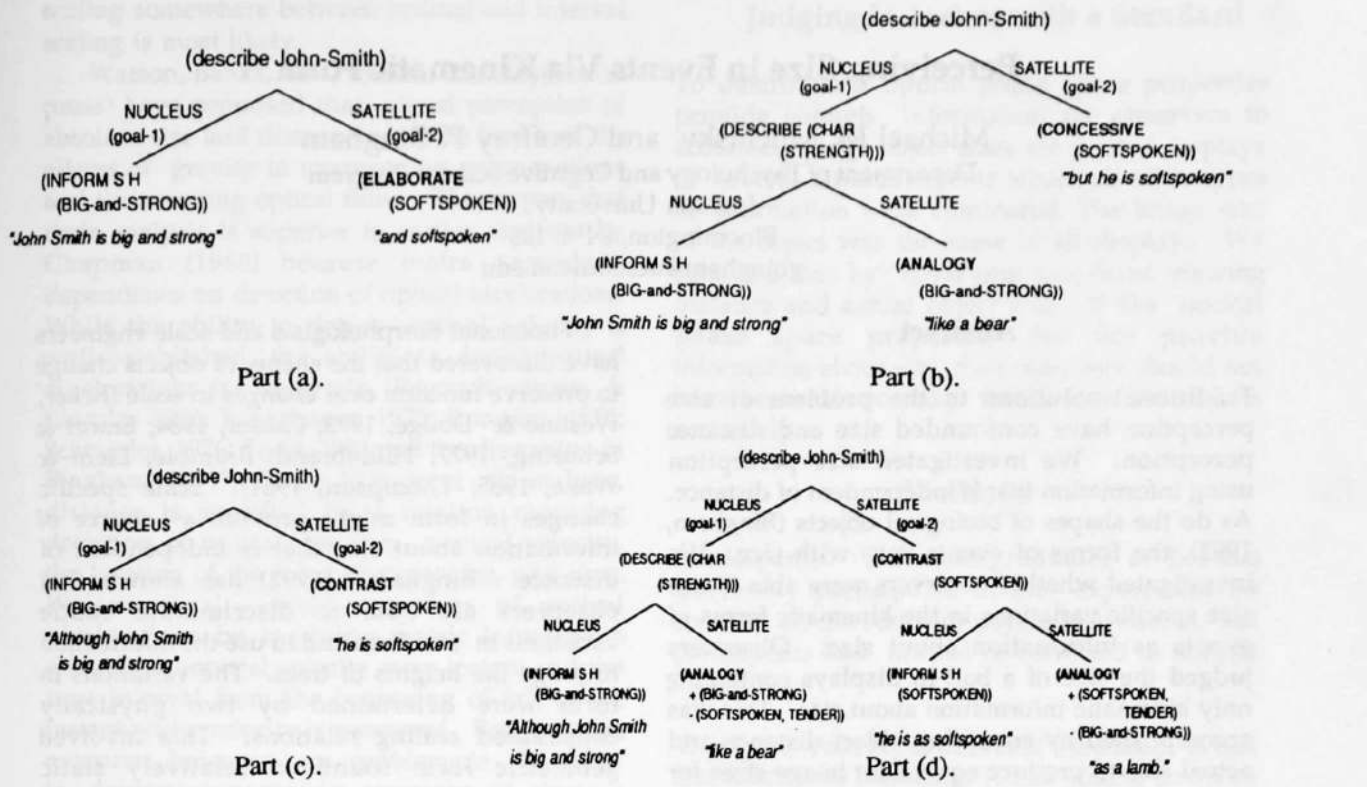

Figure 5: Skeletal fragments of the text-plan for generating the Smith texts.[5]

(Greiner, 1988) Greiner, R. (1988). Learning by Understanding Analogies. In A. Prieditis (Ed.), *Analogica* chapter 1, (pp. 1–31). Morgan Kaufmann Publishers, Inc.

(Helman, 1988) Helman, D. H. (Ed.). (1988). *Analogical Reasoning – Perspectives of Artificial Intelligence, Cognitive Science, and Philosophy*, volume 197 of *Studies in Epistemology, Logic, Methodology, and Philosophy of Science*. Boston: Kluwer Academic Publishers.

(Kedar-Cabelli, 1988) Kedar-Cabelli, S. T. (1988b). *Formulating Concepts and Analogies according to Purpose*. PhD thesis, Rutgers - The State University of New Jersey, N.J.

(MacGregor, 1988) MacGregor, R. (1988). A Deductive Pattern Matcher. In *Proceedings of the 1988 Conference on Artificial Intelligence*, AAAI.

(Mann, 1983) Mann, W. (1983). An Overview of the Penman Text Generation System. Technical Report ISI/RR-83-114, USC/Information Sciences Institute.

(Mann & Thompson, 1988) Mann, W. C. & Thompson, S. A. (1988). Rhetorical Structure Theory: A Theory of Text Organisation. In L. Polanyi (Ed.), *The Structure of Discourse*. Norwood, N.J.: Ablex Publishing Corporation.

(Moore, 1989) Moore, J. D. (1989). *A Reactive Approach to Explanation in Expert and Advice-Giving Systems*. PhD thesis, University of California – Los Angeles.

(Moore & Paris, 1989) Moore, J. D. & Paris, C. L. (1989). Planning text for advisory dialogues. In *Proceedings of the Twenty-Seventh Annual Meeting of the Association for Computational Linguistics*.

(Moore & Swartout, 1991) Moore, J. D. & Swartout, W. R. (1991). A reactive approach to explanation: Taking the user's feedback into account. In C. L. Paris, W. R. Swartout, & W. C. Mann (Eds.), *Natural Language Generation in Artificial Intelligence and Computational Linguistics* (pp. 3–48). Boston: Kluwer Academic Publishers.

(Neches *et al.*, 1985) Neches, R., Swartout, W., & Moore, J. (1985). Enhanced Maintenance and Explanation of Expert Systems through explicit models of their development. *IEEE Transactions on Software Engineering, SE-11*(11).

(Paris, 1991) Paris, C. L. (1991a). Generation and Explanation: Building an explanation facility for the Explainable Expert Systems framework. In C. Paris, W. Swartout, & W. Mann (Eds.), *Natural Language Generation in Artificial Intelligence and Computational Linguistics* (pp. 49 – 81). Boston: Kluwer Academic Publishers.

(Prieditis, 1988) Prieditis, A. (Ed.). (1988). *Analogica*. Los Altos, California: Morgan Kaufmann Publishers.

(Swartout *et al.*, 1991) Swartout, W. R., Paris, C. L., & Moore, J. D. (1991). Design for Explainable Expert Systems. *IEEE Expert*, *6*(3), 58–64.

[5]For examples of the actual text plans, including goals posted, see (Moore, 1989; Moore & Paris, 1989).