An Explanation System for Frame-based Knowledge Organized Along Multiple Dimensions

R. Gershon, Y. Ali, M. Jenkin

LCM-TR83-2 December 1983

Laboratory for Computational Medicine
Department of Computer Science
10 King's College Circle
University of Toronto
Toronto, Ontario M5S-1A4

AN EXPLANATION SYSTEM FOR FRAME-BASED KNOWLEDGE ORGANIZED ALONG MULTIPLE DIMENSIONS

by

Ron Gershon
Yawar Ali
Michael Jenkin

Abstract

This paper describes an explanation system for frame-based knowledge about events, as presented in a visual motion expert system. As such, it can be applied to representations that embody different frame organizational relationships such as is_a, part_of, instance_of, similarity, and time. This may be contrasted with most current expert systems which employ rule-based knowledge representations. In addition, such expert systems typically do not deal with complex spatio-temporal information and only a small number of them have any explanation capabilities. The system described in this document is capable of making inferences about frame comparisons and temporal relationships not present in the knowledge base, and provides output in both textual and pictorial formats. The graphical format is particularly useful for revealing the structure of the knowledge frames. The system has been implemented and tested on a knowledge base designed for human left ventricular performance assessment and examples of interaction with the system will be presented.

Acknowledgements

We wish to thank Prof. John Tsotsos for his encouragement and his helpful suggestions. His continual guidance and assistance motivated and oriented this work, and his constructive criticism and suggestions improved this document. Prof. Ray Perrault provided helpful advice on the design of the natural language interface.

Thanks are also expressed to Andrew Gullen for much programming assistance, and to Brian Nixon for his comments and advice regarding the preparation of this document.

This research has been supported in part by the Canadian Heart Foundation, the Ontario Heart Foundation, and by the Defence Research Establishment Atlantic.

Table of Contents

1 Introduction

One of the early conclusions of designers of expert systems was that they should have an explanation capability [Shortliffe76]. Although most of the systems demonstrated interesting results in their domain of expertise, there was a need to see not only correct results, but also the way by which they were achieved. In other words, the need was for tools that aid in the evaluation of the knowledge and the reasoning process of the systems. In particular, this applied to the medical domain where many computer based consultation systems were built. This domain received special attention because of the growth in medical knowledge and information and the belief that computers could serve as information stores and as analytical tools to aid in decision making. In order to increase acceptance within the medical community, and to enable users to query and verify the knowledge upon which the systems were constructed, the need to furnish the systems with explanation capabilities was recognized.

While some systems did not have any explanation capabilities at all ([Pauker76], [Pople77]), other systems included different means of explanation. In the MYCIN system [Shortliffe76], one of the primary requirements was the ability to explain its advice. The developers attribute its high level of competence to the use of modular, stylized coding, readily translatable to simple English, as a knowledge representation. The question answering capability was achieved through the retrieval of relevant rules, guided by pre-established context trees which indicate premise and action contents, or by specific events (questions and conclusions) happening during the consultation. These capabilities were transferred to the EMYCIN system [vanMelle80], with the addition of a debugging aid adapted from the TEIRESIAS program [Davis79]. Another system which produces explanations and justifications is the Digitalis Therapy Advisor [Swartout81]. This system uses an automatic programmer to generate a consulting program by refinement from abstract goals. The automatic programmer uses a domain model, consisting of descriptive facts about the application domain, and a set of domain principles which prescribe behaviour and drive the refinement process forward. By examining the refinement structure created by the automatic programmer, the system provides justifications of the reasoning process.

In this document, we present an explanation capability which was built upon a visual motion understanding system. This system, ALVEN [Tsotsos80, Tsotsos81, Tsotsos83], was designed for human left ventricular performance assessment. It is a system that combines extensive frame-based knowledge with interacting control schemes to produce an evaluation of the ventricle's condition. With its rich representation, which includes different knowledge organizations and frame inter-relationships, it is able to deal with complex issues such as time, and space. In contrast, the typical rule organization in rule-based systems is the implicit AND/OR tree formed by the premise-action pairs, and the context trees that group rules according to their applicability [vanMelle80]. Even though it may be possible to tailor such representations into dealing with complex entities, a second consideration further

compounds the difficulties inherent in rule-based systems. This consideration is that of the explanation of the knowledge. Explanation of system knowledge need not end with queries whose answers involve retrieval of chains of rules, but may also include questions that require abstraction and comparison. In the ALVEN case, abstraction can be achieved with the different knowledge organizations found in the representation, such as generalization and aggregation, while comparison is available between frames connected via similarity links.

Another feature of ALVEN which differs from other expert systems is that its control scheme does not rely on a single mechanism, but on several ones. In [Aiello83], three incarnations of the PUFF system were compared, each with the same knowledge, but different control schemes. Although the result of the comparison was that for PUFF's specific problem domain, expectation-driven (hypothesis-driven) strategy was the best, it too had drawbacks. Each of the three control schemes tested had advantages and disadvantages. For example, the expectation-driven scheme focused on important factors and came immediately to the correct conclusions, but depended heavily on the initial hypothesis, while the event-driven scheme proved to be robust and to consider all the input data, but such a scheme is known to be nonconvergent in that it can not focus on or direct the search toward a desired solution. In the ALVEN system, it was recognized early that a single control scheme would not be adequate for all situations, and thus several interacting dimensions were designed, each adding a new strength to the control process so that the disadvantages of each would be complemented by another dimension.

Clearly, the explanation capability designed for this system had to take into account the different features described above. Thus, the nature of the explanation methodology and the nature of what can be explained are different from other expert systems. We concentrated mainly on the explanation of the knowledge base. Although the explanation of reasoning was not addressed explicitly, much of it can be handled since some control information is declaratively embedded in the knowledge base.

The explanation of the reasoning process in expert systems involves recording the different intermediate computations, so that users are able to question the reasoning at any given point. Furthermore, some expert systems provide justifications to their reasoning which explain why some of their actions were made. In that manner, if the user is suspicious of the advice he receives, he is able to ask for a description of the methods employed and the reasons for employing them [Swartout81].

In ALVEN, partial results are permitted and since instance tokens are produced for each verified hypothesis, and since hypotheses maintain the organization exhibited by the frames they are formed from, interpretation results also exhibit the same structure. The existence of the hypotheses exception records, which keep track of all the exceptions raised during interpretation and the fact that the system can produce plots of the certainties of each hypothesis at any given time, are two aspects of the reasoning process

which are available to the user.

The explanation system is one that can be applied to representations that are frame-based and embody the specified organizational relationships. It is capable of making inferences about frame comparisons and temporal relationships not present in the knowledge base, and provides output in both textual format and pictorial format. It is only the first step in generating explanations from a frame-based knowledge base, and we will suggest some ideas for extending the current design in the last section.

2 The Representation Scheme

The explanation system makes great use of the different features of the representation scheme. Most of the important information which one would like to get from the knowledge base can be obtained from the organization, either directly from the different relationships, or by applying some inference rules on certain portions of the knowledge base. As will be shown, the knowledge organization permits abstractions and comparisons, which are important aspects of the knowledge explanation.

The representation of knowledge uses concepts from semantic network theory [Brachman79] and particularly the PSN formalism [Levesque79]. The basic entities of the representation are *frames* and are used to model abstract concepts. The internal structure of a frame is defined by *slots* that form its parts, each slot having a specific *type*, constraints, defaults, relations to other slots, and exceptions. The *exceptions* are frames themselves and have slots that are filled with the matching failure characteristics (the context in which the failure occurred), so that proper selection can be made of alternate frames. This selection assists in directing the system attention to other, perhaps more viable, hypotheses. Furthermore, exceptions are recorded when raised, thus showing the progress of the reasoning process at any given time. The decomposition of frames into slots defines an implicit *PART_OF* hierarchy of description. The PART_OF relationship relates a frame to its parts, which are the types of its slots. Slots are classified into two categories: *prerequisite* slots specify concepts which must be observed before the frame can be instantiated, while *dependents* slots provide additional semantic components that are included along with the frame concept on instantiation. Frames are organized along the *IS_A* relationship in order to relate general to specific frames. This provides a mechanism for imposing more global constraints on specific concepts, by means of inheritance of properties from father to son nodes in the hierarchy. Representational prototypes (frames) are distinguished from and related to tokens by the *INSTANCE_OF* relationship. Instances must reflect the structure of the frame to which they related.

In order to drive the process of selecting alternate frames, one or more *similarity links* [Minsky75] may be included with each frame. These links relate frames that are, in some sense, similar, and are used to handle situations where one or more exceptions have been raised. Similarity links have three components: source and target frames, conditions under which the comparison is valid (the "similarities"), and conditions that differentiate between concepts (the "differences"). Since only one of the frames, the source frame, is active when the exception is handled, a similarity expression is first evaluated whose predicates describe prerequisite components of the target frame that must be true that were not possessed by the source frame. On the other hand, the difference expression is a time course of exceptions which may be raised during the matching of the source frame. Each of the list's elements has an associated time interval that specifies when each exception should occur during matching. Exceptions are handled via similarity links of the PART_OF parent frame of the failing frame, which provides a context for the exception. An exception raised by a part implies that the frame itself has also failed, since PART_OF is a form of existential quantification.

Several interacting mechanisms are available for the representation of temporal information. A TIME_INTERVAL frame is defined that contains three slots, namely, start time, end time, and duration. This frame then can be included in the structure of any other frame and relations before, after, during, etc. (similar to [Allen81]) are provided. In constraint or default definition, sequences of values (or ranges of values) may be specified using an "at" ("@") operator, so that in effect a piecewise linear approximation to a time-varying function can be included. In this case, of course, constraint evaluation must occur at the proper point in time. In addition, the PART_OF hierarchy offers arbitrary event grouping in time, such as SIMUL_MOT (simultaneous motion), SEQUENCE, and OVERLAP_MOT (overlapping motion). Finally, two means of quantification are included in the representation: the "find" clause, specifying that in order to find an acceptable slot filler, the filler candidate must satisfy the given relationship with some other object in the knowledge base, and the "for all" clause, specifying that a candidate slot filler must satisfy a specified relation with all objects in the knowledge base that qualify for comparison.

3 *The ALVEN Knowledge Base*

This representation scheme was used to represent and organize motion concepts into a knowledge base. This knowledge base is part of the ALVEN system, A Left Ventricular Wall Motion Analysis System. This system utilizes a general motion knowledge base, which serves as a basis to a more specialized one -- the left ventricular knowledge base. All different components appear in the frames as slots or constraints. All motion concepts are organized along the IS_A hierarchy, which serves as a tool for specializing properties. Thus general motion concepts are found in the higher levels of the IS_A hierarchy, while specialized ones appear in the lower levels. An example of a general motion frame appears in Figure 1.

```
frame SEQUENCE is-a MOTION with
prerequisites
   motion_set : set of MOTION such that [
            for all m : (MOTION such that [
                        m element-of motion_set])
            verify [
                   m.subj = self.subj,
                   ~find m1 : MOTION where [
                        m1 element-of motion_set,
                        m1.time_int.st during m.time_int or
                        m.time_int.st during m1.time_int
                        ],
                   find m2 : MOTION where [
                      m2 element-of motion_set,
                      (m.time_int.st = m2.time_int.et or
                       m2.time_int.st = m.time_int.et )
                       ]
                 ] ,
              card(motion_set) > 1,
              strict_order_set(motion_set, time_int.st)
           ] ;
dependents
   first_mot : MOTION with first_mot <-
                          earliest_st(motion_set) ;
   last_mot : MOTION with last_mot <-
                            latest_st(motion_set) ;
   time_int : with time_int <-
              ( st of TIME_INTERVAL with st <-
                            first_mot.time_int.st ,
                et of TIME_INTERVAL with et <-
                            last_mot.time_et );

specializations : (SINGLE_BEAT exor SINGLE_INV_BEAT) or
                  (REP_BEAT exor REP_INVBEAT) or
                  (TOUCH exor SEPARATE)

end $
```

Figure 1 -- SEQUENCE, a general motion frame

Remarks
This frame describes the concept of a *sequence* of events (or motions). It looks for motions that do not occur during other motions' occurrence, but rather for motions which start as soon as their predecessor ends. The start time of the sequence is defined as the start time of the first motion and the end time as the end time of the last motion.

The system accepts as input digitized X-ray films of the human left ventricle. The image sequences that are being studied are obtained by cineradiography. A film is taken at 60 images per second, and these images are then inspected by cardiologists for the purpose of evaluating the patient's condition. Using information about the LV performance in different cases, as well as a large number of specific data for a variety of disease states, a

problem-specific knowledge base was constructed. A frame which describes a left ventricular phase can be found in Figure 2.

```
frame N_RAPID_EJECTION is-a VOL_UCONTRACT with
prerequisites
   subj : N_LV ;
   motion_set : such that [
                 card(motion_set)=3
                ] ;
   anterior : N_ANT_RAPEJ such that [
                 anterior element-of motion_set
              ] ;
    apical : N_AP_RAPEJ such that [
               apical element-of motion_set
             ] ;
    posterior : N_POST_RAPEJ such that [
              posterior element-of motion_set
            ] ;

dependents
   time_int :  with time_int.dur <- default(0.09*(60/(0.8*HR)));
   vrate : with vrate <- (start_y - end_y)/IFI default(690)
           such that [
               vrate > 420 /* 4 */
               exception [TOO_SLOW_RAPID_EJECTION],
               vrate < 960 /* 4 */
               exception [TOO_FAST_RAPID_EJECTION]
               ] ;

similarity links

link1 : HCM_RAPID_EJECTION
(for differences :
   d1 : HYPERKINESIS ; /* 4 */ )

end $
```

Figure 2 -- N_RAPID_EJECTION, a domain specific frame

Remarks
The frame describes the normal rapid filling phase of the left ventricular contraction. It is a simultaneous motion of the three segments of the left ventricle, which contract towards the centroid. Its default duration is 0.09 milliseconds (normalized to the patient's heart beat). The volume rate is calculated by dividing the difference in volume at start time and start time by the inter-frame interval (IFI = 1/60 for 60 frames per second). Abnormalities in the volume rate in this phase activate two different exceptions (TOO_FAST or TOO_SLOW_RAPID_EJECTION).

4 Classes of Knowledge Queries

The main effort of this research was concentrated on generation of the correct answers to different questions. As mentioned before, the questions evolve from the structure of the knowledge base. We therefore distinguish between two main aspects of the explanation capability in the system: explaining the contents of specific frames, and explaining the organization of frames. One important note is the fact that it is expected that the system will be used by different users. Therefore, there will be some variations to questions, depending on the users' background or knowledge of the system. In this section, we shall try to show some examples of questions as asked by different users. All the examples will be of "canonical" questions, i.e., questions in their extended form, without complex syntax or abbreviated wordings. For some examples of variations in wordings, see Appendix A.2.3.

4.1 Presenting contents of frames

The system has the ability to extract portions of information found within frames, as requested by the user. In this manner, any part of the frame definition is exposed for purposes of study, consultation, or modification. Exact definitions of each frame can be presented to the user, revealing the actual quantitative and qualitative data. Some examples of questions, as they can be asked by someone who understands the system (such as a knowledge engineer) are :

- What are the parts of frame F?
- What are the constraints on slot S?
- On instantiation of this frame, how is the value for slot S determined?
- What exceptions can be raised by slot S?
- If exception E is raised, how are its slots to be filled?

Since domain users do not know about the frame structure and components, and they are likely not familiar with the existence of exceptions, constraints, etc. -- therefore a rephrasing of the above canonical questions accepted by the system and appropriate for such users is:

- How is F defined?
- What are the characteristics of S?
- How is S computed?
- What are the possible anomalies in S?
- What are the important aspects of E?

To illustrate what kind of information is obtained, Figure 3 presents two examples of explanations concerning contents of frames.

```
>>> show me the constraints on slot SUBJ in N_SYSTOLE


          subj : N_LV such that [
        find narrow : NARROW where [
           narrow.subj = self.subj ,
           narrow.time_int during self.time_int,
           narrow.speed < 150  and narrow.speed > 50 /* 1 */
          ]
         exception [NARROWING_IMPROPERLY]
       ] ;



>>> how are the slots  of  exception  TOO_MUCH_MOTION  to  be
    filled in the slot EXTENT of the frame HCM_POST_DIASTASIS


   subj : PHYS_OBJ ;
   time_int : TIME_INTERVAL ;
   source_type : FRAME ;
   source_id : INTEGER ;

   disp : LENGTH_VAL with disp <- dist(subj.centroid @
                                     source_id.time_int.st,
                                   subj.centroid @
                                     source_id.time_int.et ) ;


    seg <- "posterior"
```

Figure 3 -- Extracting contents of frames

Note that in the first example of Figure 3, there is a number between two asterisks on the right hand side (i.e., " /* 1 */ "). This points the user to a certain reference (from the medical literature) which was used by the knowledge engineer when the knowledge base was constructed. This reference can be queried by the user, as shown in Figure 4.

```
>>> what is reference number 1 ?

1. Doran, J., Traill, T., Brown, D., Gibson, D., "Detection of
   abnormal left ventricular wall movement during isovolumic
   contraction and early relaxation", BRITISH HEART JOURNAL,
   Vol. 40, 1978.
```

Figure 4 -- Asking for a reference used in the knowledge base

4.2 Presenting frame organization

The second set of questions are "organization" questions. The ability to traverse along the different organizations allows the user to obtain a general picture of the knowledge base, without getting into detailed descriptions (which are the contents of the frames). Therefore one can see how certain concepts are specialized, how complex concept aggregations are broken down into simpler ones, how the failure of one hypothesis leads to the consideration of a different one, and how events are ordered in time. This constitutes the four dimensions along which one can trace information: the IS_A relationship, the PART_OF relationship, similarity links, and temporal connections. Some examples of organization questions are:

- Show me the IS_A hierarchy rooted at frame F.
- What are the direct PART_OF descendents of frame F?
- Show me the similarity relationships that event E has.
- What are the differences between frames E and F?
- Show the temporal connections between E and F.
- What event precedes event E?

Some analogous questions for users not familiar with the system's structure are as follows:

- What kinds of F's do you know about?
- Show me all the components of the definition of F.
- Show me the possible alternatives to E.
- What are the differences in diagnostics between E and F?

The following sections will discuss separately the questions that deal with similarity links and time, since they involve more complex problems. The questions that involve the IS_A and PART_OF relationships involve traversal of directed graphs (the IS_A and PART_OF hierarchies). This information is presented using graphics package since it is more convenient to see it in pictorial form. All frames have both IS_A and PART_OF relationships

with other frames. Thus, the level of specifity of detail can be controlled by, or examined by traversing, the IS_A hierarchy, while the level of resolution of detail (decomposition) is reflected in the PART_OF hierarchy. In [Patil82], only decomposition view of level is present, while in CADUCEUS [Pople82], the level of specifity is employed and level of resolution is restricted to causal connections. An example of an IS_A question can be found in Figure 5.

```
>>> display IS_A ancestors of VOL_UCONTRACT
```

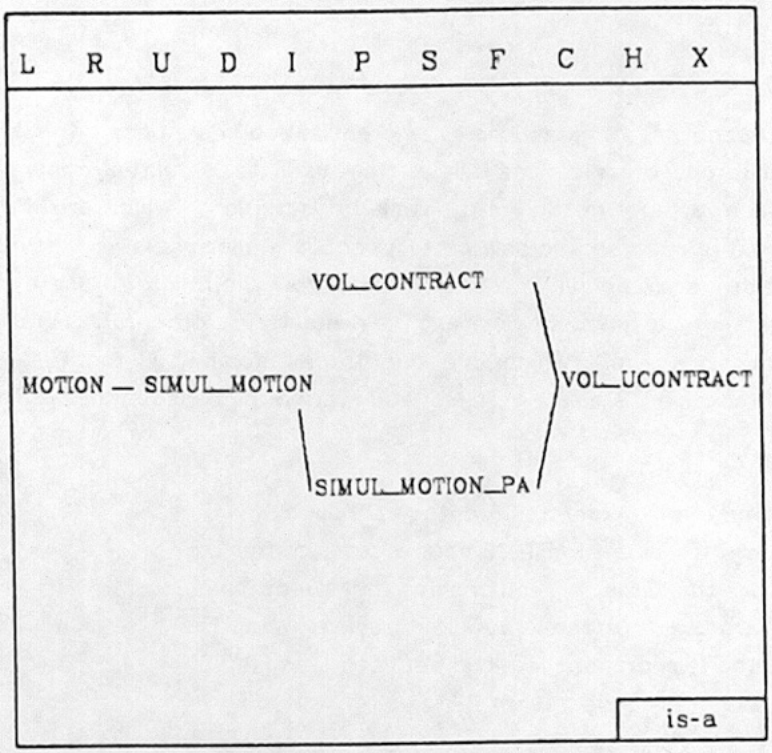


Figure 5 -- A query involving the IS_A hierarchy and its answer

Remarks
This diagram does not reveal all the is-a relations of certain frames. The ancestors of VOL_UCONTRACT are as shown, but their ancestors are not necessarily complete. With the help of the menu (see Appendix A.3), one can obtain more information about any of the frames.

5 Frame Comparisons

Similarity links connect concepts which have both some common features and some differences, and are considered as competitors in a recognition process. For further details on the control scheme in ALVEN see [Tsotsos80, Tsotsos83]. If the user asks about frame comparisons, the explanation module must show how the frames differ and how they are similar. The significance of this comparison is the fact that the similarity links are part of the control scheme. Being able to see how and why control is transferred from one hypothesis (frame) to another allows the user to understand what options the system has in case certain hypotheses fail.

Initially the system has to find out whether the frames in question are comparable, i.e., check if they are connected via similarity links. There does not have to be a direct connection -- there could be a chain (or chains) of links connecting two frames.

The next step performed by the system depends on the type of information sought. As mentioned in Section 2, there is a distinction between the similarity expression and the difference expression. An example of a similarity link with these expressions appears in Figure 6.

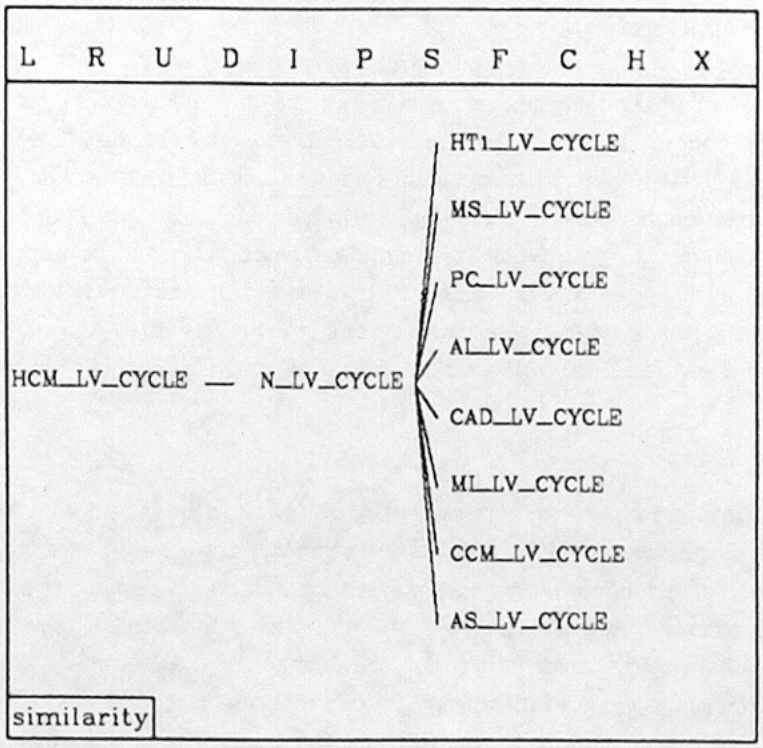
```
frame N_LV_CYCLE is-a SEQUENCE with
   prerequisites
     :
     :
   dependents
     :
     :
   similarity links
     :
   link7 : HCM_LV_CYCLE
   with similarities :
      iso_contract instance-of N_ISO_CONTRACT ;
   (for differences :
      d1 : MINAXIS_TOO_SHORT where [
           d1.time_int = systole.time_int.et  ]  ;
      d2 : MINAXIS_TOO_SHORT where [
           d2.time_int = iso_relax.time_int.st ]  ;
      d3 : TOO_LONG_ISORELAX ;
      d4 : TOO_LOW_ESV ;      )
end $
```

Figure 6 -- One of the similarity links in the frame N_LV_CYCLE

Because of this distinction, for the purposes of explanation the system treats the common features and differences differently: common features are found between chains of frames and are presented as they appear in the "with similarities" component; differences can be compared along a chain of frames, but comparison is conducted only if the same exceptions are involved. The reason is that these exceptions are the driving force that causes one frame to activate another, and following an exception along the similarity chain is a way of examining how one phenomenon can cause the activation of different alternatives. Consequently, if there exists a similarity path of frames with no common exceptions, the result of such a search will be regarded as failure -- no common thread of differences. However, if the system does find a path with common exception types, the constraints attached to the exceptions are examined. If they do not involve the same slot fillers (subjects, referents, etc.) -- the system notes the fact that the exception types themselves were common but not their tokens. On the other hand, if the constraints do involve common slot fillers, the system finds which ones form the difference chain. Figure 7 presents an example of finding differences between frames.

```
>>> What are the differences between HCM_LV_CYCLE and
                                                    CCM_LV_CYCLE ?

 THE SIMILARITY CHAIN BETWEEN THE TWO FRAMES CAN BE OBSERVED
 ON THE SCREEN.
```




```
                    THE DIFFERENCES ARE:

TOO_HIGH_ESV
-----------------------

MINAXIS_TOO_LONG
-----------------------

        d3.time_int = systole.time_int.et
        d2.time_int = diastole.time_int.et
```

Figure 7 -- Finding differences between frames

Remarks
In this example, there is a chain of frames leading from HCM_LV_CYCLE through N_LV_CYCLE

to CCM_LV_CYCLE. Thus, for every difference common to the frames (excluding the destination -- CCM_LV_CYCLE), the response accumulates information from all the frames along the chain. Here, two differences are observed : TOO_HIGH_ESV (end systolic volume), which has no additional constraints attached to it, and MINAXIS_TOO_SHORT (the small axis is too short), and the constraints are as shown.

6 Explanation of Temporal Information

Perhaps the most challenging aspect is the explanation of temporal information. The reason is that some concepts inherit their temporal properties from the IS_A hierarchy (if CONTRACT is_a SIMUL_MOT, then all its component events occur at the same time), while other temporal information is found within slots (in particular, "time_int"), and within "at" clauses. All frames in our system which are motion frames (i.e., not object frames) are events. Even all the derivatives of the NO_MOTION frame are, in fact, events which did not happen. The task of the system is to take these frames as they appear within the knowledge base and map them onto a time scale according to their relative occurrence.

6.1 Assumptions

The ALVEN system contains a general motion knowledge base, as well as a domain specific motion knowledge base (left ventricular motion concepts), the domain knowledge being defined in terms of the general knowledge. General motion concepts can be thought of as *temporally unbound motions* while domain specific motion concepts as *temporally bound motions*. This distinction was made in order to emphasize that although some motion concepts may be the same, we can usually place quantitative temporal restrictions on the motions in the application domain, while general motion concepts remain always unbound quantitatively. For example, N_LV_CYCLE (normal left ventricular cycle) is N_SYSTOLE followed by N_DIASTOLE (which is the contraction phase in the heart followed by expansion) while BEAT is CONTRACT followed by EXPAND. The former, however, has values of time intervals defined for each phase while the latter does not. Ordering bound and unbound motions uses the same process -- the difference is that bound motions can be mapped onto a specific time scale.

Finally, the design of the explanation capability relies on the fact that there are frames such as SEQUENCE, SIMUL_MOT, and OVERLAP_MOT which exist and define temporal grouping of sets of events.

6.2 Temporal Ordering

For the purpose of relating events to each other, the notation suggested in [Allen81] is used (relations such as "after", "is met by", etc.). The process of ordering is facilitated by the fact that frames contain all the information (including the temporal ordering) about their immediate descendents. Figure 8 is used to illustrate this point. In N_LV_CYCLE we can learn that N_ISORELAX is met by N_SYSTOLE , so when the traversal reaches the next level, the system already knows that N_SLOW_EJECTION must be met by N_ISORELAX. Therefore, information is local to the frame itself, but it is global to all the components of that frame.

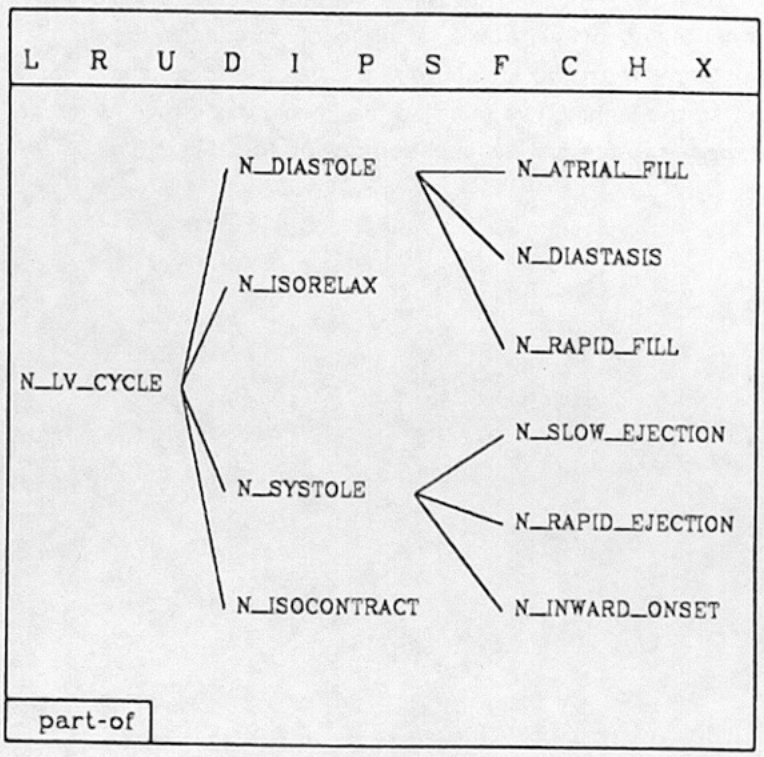
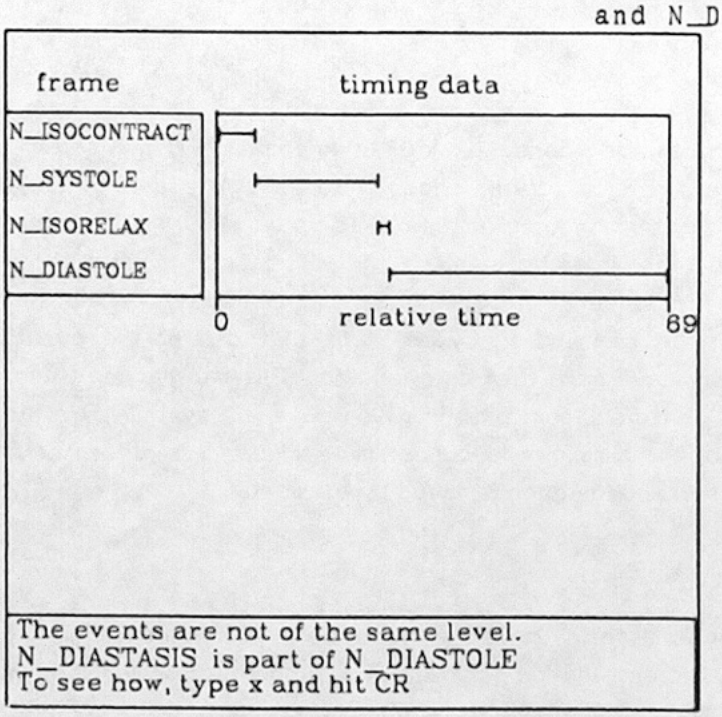


Figure 8 -- An example from the PART_OF hierarchy in the knowledge base

In order to demonstrate the method by which the system orders frames, we shall use one of the questions presented in Section 4.2. The problem will be to show the temporal connections between two bound motions. Suppose these two events are N_DIASTASIS and N_SYSTOLE, and they appear in the PART_OF hierarchy presented in Figure 8.

The first problem the system has to tackle is the fact that the two events are not in the same PART_OF level. It is semantically more appropriate to relate the event of the higher level (in our example -- N_SYSTOLE) to the other one's PART_OF ancestor that shares the same level in the hierarchy (N_DIASTOLE). The lower level event (N_DIASTASIS) appears within its ancestor's (N_DIASTOLE) context. Since the PART_OF hierarchy enables the system's designer to break complex concepts into simpler ones, what this means is that the comparison will be performed between concepts of the same complexity, and then, further information will be provided to show where and how the simpler concept appears within the more complex context. To illustrate this point in a different domain, consider the comparison of an airplane taking off with the lowering of its landing gear. It is more meaningful to compare the take-off to the landing process, and indicate that that among the processes occurring during the landing, we can find the lowering of the landing gear. An example of the system's response to the original question can be found in Figure 9.

>>> show me the temporal connections between N_SYSTOLE
and N_DIASTASIS



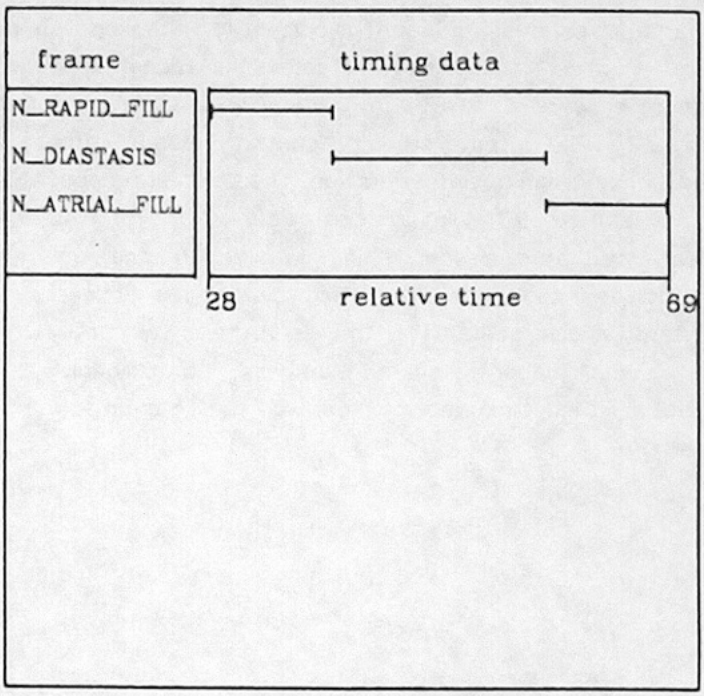


Figure 9 -- Relating motions in time

Remarks
The upper figure presents the temporal ordering between the higher level events, while the

figure below provides more details about the lower level event's place within its ancestor's context.

The process of ordering is done in two phases: (1) the system has to search up the PART_OF hierarchy for a common ancestor to the two events in question, which will be an event containing them or their ancestors as parts; (2) from the common event, the system traverses down the hierarchy, level after level, relating the parts according to their temporal ordering. (The detailed algorithm appears in [Gershon82]). The search up the PART_OF hierarchy can be complicated by the fact that one of the events in question, or its ancestors, may have more than one ancestor. This can happen when an event appears in two (or more) different contexts, in which case the system asks the user to resolve the ambiguity. For the simplicity of our example, Figure 8 is presented as a tree, i.e., all the ambiguities have already been resolved by the user.

7 Implementation

The general design of the explanation system consists of three basic steps -- an input module, the explanation generator, and an output module (see Figure 10). The purpose of the input module is to enable the user to ask questions in natural language, with some restrictions, and to activate the corresponding explanation function within the generator. Therefore, a parser was designed and implemented, enabling the system to cope with questions asked in free format and different variations. Methods employed by the explanation generator were discussed in Sections 4-6 while implementation details can be found in [Gershon82]. The output module presents the sought information in a suitable form, either using textual or graphic representation. The textual response is either the contents of frames as they appear in the knowledge base, or lists of frames which share common features as requested by the user, along with some additional remarks. The graphical representation includes tools for presenting hierarchies or parts of them, as well as displaying events on a time scale after they have been temporally ordered by the system. In this section, we shall elaborate on the input and output modules, giving a description of their features and abilities. Description of the way to use them appears in Appendix A.

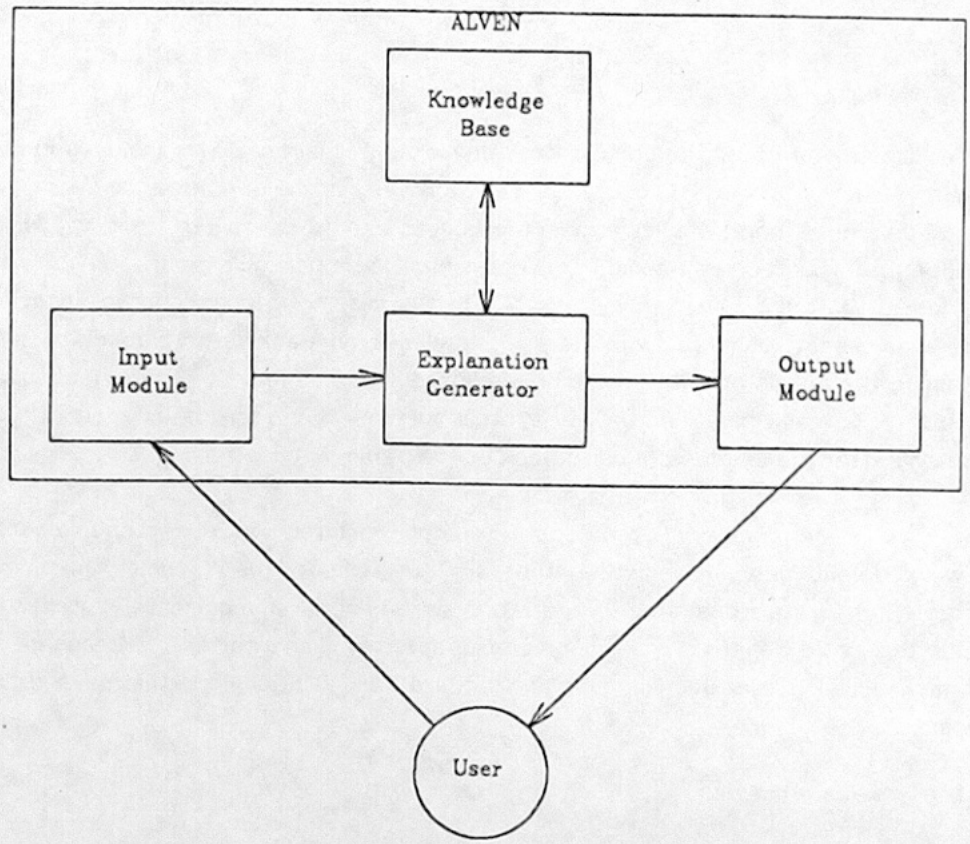


Figure 10 -- General sketch of the explanation system

7.1 The parser

The parser written for the ALVEN explanation system utilizes a semantic grammar in order to efficiently recognize English questions dealing with the organization and contents of the system's knowledge base. The grammar is represented by means of Augmented Transition Networks (ATNs) and the parser makes use of an ATN interpreter originally written by Hector Levesque in UCI/RUTGERS/TORONTO LISP for the DEC-10, later modified for FRANZ LISP running on a VAX-11/780 by Yawar Ali.

Effort has been made to permit the user a considerable degree of freedom in choosing the syntax and wording of queries, within the limitations inherent in the domain-specific semantic grammar. Provision has been made for accommodating variations in the tense and number features used in the expression of queries by the user. The dictionary employed by the system is kept compact by avoiding the explicit inclusion of lexical variants, wherever possible. Instead, a heuristic algorithm (due to [Winograd72]) is employed to analyze and modify word endings in an attempt to transform unknown words to known ones, under the assumption that they conform to the regular patterns of variation found in English. Irregular variants of words are explicitly included in the dictionary. In many instances, passive voice forms are also accepted.

Ellipsis can be employed to almost any extent desirable. This includes curtailment or omission of the preamble to a query, as well as brevity in the specification of the information that is desired. There is also some provision to handle so-called "semantic ellipsis": for instance, if a user requests some information involving a particular frame, but fails to specify the identity of the frame, the system fills in the missing information by "guessing" which frame was meant, using a heuristically-guided search backwards through a context list containing the names of entities mentioned in previous queries. The same mechanism is used to resolve pronoun reference. The system informs the user of any substitutions and insertions after the successful completion of the parsing phase for a query.

The present version of the natural language interface is sentence-oriented. This allows efficient query-recognition but fails to achieve the flexibility and robustness desirable in a system designed to operate in more general environments. Future versions of the system could benefit from a discourse-oriented English interface, allowing more flexible and natural dialogue taking account of standard conversational postulates and contextual information.

7.2 The graphics routines

There are two main graphic routines that create the different displays used by the system: TREE and TRP. Both produce a display intended for a colour raster device, with provisions to produce a hardcopy listing of each display on the Versatec plotter.

7.2.1 TREE

TREE was designed to allow a user to explore, in an interactive manner, the is-a and part-of hierarchies, and the similarity network present in the knowledge base. Within each of the hierarchies a node can be considered to form the root of two trees; a tree of it's ancestors, and a tree of it's descendents. To display the hierarchical relationships of a node, we need only know which hierarchy to examine and in what direction to expand.

The TREE program uses a tablet to allow the user to point to the nodes he wishes to examine on a colour raster display. A set of operations can be performed on the display by pointing to certain labeled regions of the display using a cursor. These options allow the user to change hierarchies, change the direction of search, and to redefine the node to be used as the root of the tree. Other options include the ability to scroll through a list of nodes if it is not possible to display all of the sons of a given node on the screen. The TREE program is capable of displaying a few generations of ancestors or descendents for a given root, examining only one son at each level. An option is provided to choose which son will be "favoured" by having his sons examined at the next level. For a detailed description of the different operations allowed by TREE, see Appendix A.3.

7.2.2 Temporal Relationship Plot (TRP)

As TREE was designed to explore graphically the is-a, part-of hierarchies, and the similarity network, present in the knowledge base, TRP was designed to examine the temporal connections. This information is not present explicitly in the knowledge base, as are the is-a, part-of, and similarity connections, rather it is present implicitly in terms of temporal restrictions present in certain frames. The problem of determining the temporal relationships between frames is solved by another part of the explanation system, namely, the explanation generator. TRP only displays the results obtained by this earlier processing.

TRP expects that the frames it has been requested to display have been related by the temporal primitives, equal, before, after, meets, met-by, overlaps, and overlapped-by. Each frame is permitted a duration, and a variance value. If no value is supplied a default value is generated. In addition, some of the temporal primitives are permitted to be quantified. For example, frame A occurs 5 seconds before frame B. If no quantification is present then default values are chosen.

8 Summary

The explanation capability described in this paper provides an interactive mechanism for user communication with the knowledge base. It supplies tools with which the user can be exposed to the knowledge base and gather, study, or query its information. It was built upon a knowledge base in which each concept was represented as a frame, and all frames were linked through different relationships. These relationships enable the system to cope with issues such as time, space, and events, whereas other systems, in particular, the rule-based ones, do not offer adequate solutions to these complex issues. Moreover, the explanation of the knowledge base need not end with queries whose answers involve retrieval of chains of rules, but rather handle a variety of cases, such as abstraction along one of several directions (e.g., IS_A, PART_OF), temporal ordering of events, comparisons between different frames, and the ability to extract frames and parts of them. Since the explanation system makes use of the underlying representation, and generates the explanations without using domain specific constructs, it can handle any knowledge base organized similarly. The only restriction is that if one wants to explain temporal connections, one has to assume the existence of some concepts (as explained in Section 6.1). Another important characteristic of this system is the fact that it combines text and diagrams in the explanation process. It is clear that some explanations are easier to understand through diagrams and therefore the graphics package was included in the system. This increases the system's ability to present hierarchies, networks, and temporal ordering.

9 Future Work

The goal of this research was to build a system capable of explaining a multiple dimension knowledge base. This task, the results of which were described in this paper, has been the first phase of building a complete explanation system, one that will explain the knowledge base and the reasoning process. Therefore we see as a natural extension of the current version the development of explanation of the reasoning process. Unlike rule-based systems, where this task is performed by back-chaining and rule retrieval, the task in the ALVEN system is more complicated because of the different dimensions of the representation scheme and the different control mechanisms it employs. Nevertheless, some of the reasoning can already be explained using mechanisms described in this document due to the declarative nature of some of the control mechanisms, namely, similarity links and exception tokens.

The combination of textual and pictorial output from the system proved to be important, and can serve as a basis to future expansion. In this respect, the natural language interface, and in particular, the ATN, can be expanded to handle the tablet buttons as noun phrases specifying location on the screen. Since the process of explanation between humans involves pictures, diagrams, and many references to pictures ("hand waving"), we think that including similar features in the explanation system will make the communication between the system and the user more pleasant. Thus, when the user points with the tablet to a certain object on the screen, the system will be able to understand what is the concept behind that object.

Another aspect of the pictorial output is creating knowledge-based animation for certain questions [Drewery84]. Since the ALVEN system employs projections to create predictions for the low level image operators, it is able to project hypotheses into signals. Therefore, when a question such as "What does a normal left ventricular (LV) cycle look like?" is asked, the word "look" can be interpreted as "the system's definition of the LV cycle" and thus the response can be the frame NORMAL_LV, or it can be interpreted literally, which will cause an animated diagram of the left ventricle "beating" to appear. This issue involves solving problems in natural language understanding ("what does the user mean"), as well as the problem of creating the images (diagram, picture) out of the knowledge base (see [Zeltzer83]).

Finally, we suggest that the knowledge acquisition issue should be addressed, so that the users will be able not only to see what is in the knowledge base, but actually modify it according to their understanding of the domain. In the same way that other knowledge acquisition systems were proposed (such as TEIRESIAS [Davis79]), the system should enable the user to check the information in the knowledge base with the tools that were presented in this paper, and when necessary, make the corrections in the appropriate locations. That will create an environment in which one can be satisfied with the contents of the knowledge base, a factor which will increase the system's acceptance among the users.

APPENDIX A
User Manual

A.1 General

The explanation system has been implemented on the CSRG VAX 11/780, running Berkeley Unix 4.1c. It is running as part of LISP. In order to activate it, the user has to enter the Franz Lisp interpreter and load some files. The explanation system resides in

/u5/alven/explain

After switching to this directory, the user is required to enter Lisp by typing "lisp". As soon as Lisp responds (prompting with "->"), the loading of the explanation system is performed by typing "[include setup]", and after the loading phase is completed (the system prompts again with "->"), the explanation system can be called by typing "[explain]". Once in the explanation system, the user can ask any question by typing it after the explanation system's prompt (">>>"). Help is available by typing "help", and quitting the explanation session is achieved by typing "quit".

A.2 The natural language interface

A.2.1 Some notes on the parsing process

Upon entry to the explanation system the user is prompted to enter a question. The English input is read in character by character, discarding punctuation symbols, and split up into words. Once this has been done, dictionary lookup is performed for each of the words entered, with substitutions being made for inflected words and synonyms. In case an unknown word is encountered it is subjected to a number of transformations, including mapping between upper case and lower case, and the previously-mentioned algorithm of Winograd. Failure of all these attempts results in the user being requested to rephrase the query, after being shown a list of the words that were not recognized by the system.

Successful completion of the lexical analysis phase is followed by an attempt to parse the query. The input is first scanned for keywords to determine the order of application of the ATN subnets. This usually results in a considerable reduction in the amount of backtracking done by the parser. The semantic grammar takes advantage of the restricted domain of discourse, allowing the user the flexibility to make use of abbreviatory syntactic devices. Recognition of a query results in a direct call to the particular explanation module designed to respond to it, followed by an opportunity for the user to

continue the dialogue by entering another question.

A.2.2 Types of acceptable queries

The natural language front end of the explanation system is capable of parsing the following canonical queries:

1. ISA questions --- requests to display one of the following:

- The isa hierarchy.
- The isa hierarchy rooted at a given frame.
- The isa ancestors of a given frame.
- The immediate isa ancestors of a given frame.
- The isa descendants of a given frame.
- The immediate isa descendants of a given frame.

2. PARTOF questions --- requests to display one of the following:

- The partof hierarchy.
- The partof hierarchy for a given frame.
- The hasaspart hierarchy for a given frame.
- The branch of the partof hierarchy between two given frames.
- The parts of a given frame.
- The immediate parts of a given frame.
- The frames that a given frame is a part of.
- The frames that a given frame is an immediate part of.

3. Time questions --- queries regarding the following:

- The temporal relationships of a given event.
- The events that precede a given event.
- Whether a given event precedes another given event.
- The events that follow a given event.
- Whether a given event follows another given event.
- The events that occur simultaneously with a given event.
- Whether two given events occur simultaneously.
- The temporal connections between two given events.
- The sequence of events that defines a given event.

4. Similarity questions --- queries regarding the following:

- Whether two given events can be compared.
- The differences between two given events.
- The similarity network.
- The similarity relationships of a given event.
- The similarity relationships a given event can inherit.
- Whether a given event can be activated via similarities from another given event.

5. Frame internals questions --- queries concerned with the following information about the internals of a frame :

- The constraints on a given slot.
- The exceptions that can be raised from a given slot.
- How the value of a given slot is determined on instantiation of the frame.
- How the slots of a given exception are to be filled in a given slot.
- Whether there is a temporal dependence.
- The display of the entire frame.
- The display of the dependents part.
- The display of the prerequisites part.
- The display of the similarity links part.

6. Miscellaneous questions.

- The name of the source referred to by a given reference number.

A.2.3 Some examples of questions

Here are some examples of queries and a few variations for each query. The extended form of the queries is presented first (in *italics*), followed by some of the acceptable alternative wordings for each query.

>>> *Show me the sequence of events that defines event m.*
>>> Display sequence of events that defined event m.
>>> sequence of events defining event m.

>>> events that define m.

* >>> events defining that event.

* >>> What are the events which define it?

* >>> Events defining it.

--- --- ---

>>> *What are the frames that are a part of frame m?*

>>> Show the frames that were part of m.

>>> which are the parts of m?

* >>> Frames which are partof the frame?

* >>> parts of it?

--- --- ---

>>> *Does event m have any features in common with event n?*

>>> Do events m and n possess any common features?

* >>> Are there any properties in common between the frames?

* >>> Does it share some properties with n?

* >>> Properties shared between them.

* Assuming an appropriate discourse context has been established by a previous query.

A.3 How to use the graphics package

Using a tablet the user can position the cursor on the screen. At the top of the display a series of boxes are displayed, each marked by an identifying letter (see Figure 11). By moving the cursor over one of the boxes and depressing one of the buttons on the tablet the function associated with that box will be executed. The following functions are supported

L Scroll left. A row with the word MORE appearing to the left of the display can be scrolled left to display more nodes. Point at the row to scroll and depress the

button.

R Scroll right. Similar to "L" above.

U Make the current direction up. Point at the node you wish to form the root of the new tree.

D Make the current direction down. Similar to "U" above.

I Make the current hierarchy the is-a hierarchy.

P Make the current hierarchy the part-of hierarchy.

S Make the current hierarchy the similarity hierarchy.

F Point to the node to favour by expanding his sons.

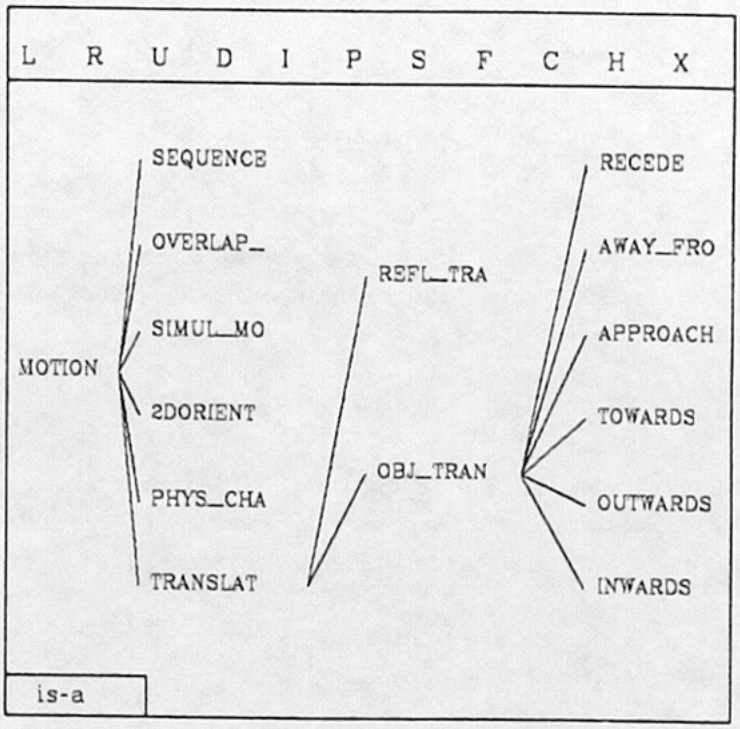
H Produce a hardcopy of the display on the Versatec printer/plotter. Figure 11 was produced using this feature. Note that due to restrictions of the Versatec the full frame name will not be printed on the hardcopy.

C Rather than pointing at a node or at a row, a command may be canceled by pointing at this function.

R Return to the calling program. Point at the node to return.

X Exit to the system. No return value is required.

Figure 11 -- A display of the screen



References

[Aiello83] Aiello, N. "A comparative study of control strategies for expert systems: AGE implementation of three variations of PUFF", Proc. AAAI-83, 1983

[Allen81] Allen, J.F. "Maintaining knowledge about temporal intervals", University of Rochester - Dept. of Computer Science TR-86, 1981

[Brachman79] Brachman, R.J. "On the epistemological status of semantic networks", in *Associative networks: Representation and use of knowledge by computers* (Findler, N.V., ed.). Academic Press, New York, 1979

[Davis79] Davis, R. "Interactive transfer of expertise: Acquisition of new inference rules", *Artificial Intelligence 12*, 1979

[Drewery84] Drewery, K. "Knowledge based animation as part of explanations for expert system knowledge", M.Sc. Thesis, Dept. of Computer Science, University of Toronto, in preparation

[Gershon82] Gershon, R. "Explanation methods for visual motion understanding systems", M.Sc. Thesis, Dept. of Computer Science, University of Toronto, 1982

[Levesque79] Levesque, H., and Mylopoulos, J. "A procedural semantics for semantic network", in *Associative networks: Representation and use of knowledge by computers* (Findler, N.V., ed.), Academic Press, New York, 1979

[Minsky75] Minsky, M. "A framework for representing knowledge", in *The Psychology of Computer Vision*, (Winston, P., ed.), McGraw-Hill, 1975

[Patil82] Patil, R., Szolovits, P., and Schwartz, W. "Modeling knowledge of the patient in acid-base and electrolyte disorders", in *Artificial Intelligence in Medicine*, (Szolovits, P., ed.), Westview Press, 1982

[Pauker76] Pauker, S.G., Gorry, G.A., Kassirer, J.P., and Schwartz, W.B. "Toward the simulation of clinical cognition: Taking present illness by computer", *The*

American Journal of Medicine 60, 1976

[Pople82] Pople, H. "Heuristic methods for imposing structure on ill-structured problems: The structuring of medical diagnostics", in *Artificial Intelligence in Medicine*, (Szolovits, P., ed.), Westview Press, 1982

[Pople77] Pople, H.E. "The formation of composite hypothesis in diagnostic problem solving: An exercise in synthetic reasoning", *Proc. of the Fifth International Joint Conference on Artificial Intelligence*, 1977

[Shortliffe76] Shortliffe, E.H. *Computer based medical consultation: MYCIN*. Elsevier North Holland Inc., 1976

[Swartout81] Swartout, W.R. "Producing explanations and justifications of expert consulting programs", M.I.T. Laboratory for Computer Science TR-251, 1981

[Tsotsos83] Tsotsos, J.K. "Representation axes and temporal cooperative processes", in *Vision, Brain and Cooperative Computation*, ed. by M. Arbib and A. Hanson (in press)

[Tsotsos81] Tsotsos, J.K. "Temporal event recognition: An application to left ventricle performance", *Proc. of the Seventh International Joint Conference on Artificial Intelligence*, 1981

[Tsotsos80] Tsotsos, J.K. "A framework for visual motion understanding", Ph.D. dissertation, Technical Report CSRG-114, Dept. of Computer Science -- University of Toronto, 1980

[vanMelle80] van Melle, W.J. "A domain independent system that aids in constructing knowledge based consultation programs", Ph.D. Dissertation, Stanford University -- Dept. of Computer Science, STAN-CS-80-820, 1980

[Winograd72] Winograd, T. *Understanding Natural Language* Academic Press, New York, 1972

[Zeltzer83] Zeltzer, D. "Knowledge-based animation", Proc. ACM SIGGRAPH/SIGART Workshop on Motion: Representation and Perception, April 1983