A Goal-Directed Hearsay-II Architecture:

Unifying Data-Directed and Goal-Directed Control

Daniel D. Corkill and Victor R. Lesser

Computer and Information Science Department
University of Massachusetts
Amherst, Massachusetts, 01003

COINS Technical Report 81-15

June 1981

ABSTRACT

A number of limitations in the purely data-directed approach to control taken in Hearsay-II have become apparent as the architecture has been applied to non-speech task domains. These limitations stem from the inability to plan sequences of knowledge source (KS) executions and to instantiate KS activity based on criteria other than data-directed events. By introducing goal-directed control into the Hearsay-II architecture, more sophisticated forms of control can be implemented. Goal processing is introduced into the architecture by splitting the mapping of data-directed events -> KSs into two mappings: events -> goals and goals -> KSs. The events -> goals mapping explicitly represents goals that are implicitly satisfied in the data-directed system by KS execution. In order to process goals, the Hearsay-II architecture is augmented to include a parallel blackboard, called the goal blackboard, and a new control component, the planner. The augmented architecture unifies data- and goal-directed activity and permits sophisticated forms of control to be implemented.

This research was sponsored by the National Science Foundation under Grant MCS-8006327 and by the Defense Advanced Research Projects Agency (DOD), monitored by the Office of Naval Research under Contract NR049-041.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the National Science Foundation, the Defense Advanced Research Projects Agency, or the U.S. Government.

1.0 Introduction

In a retrospective article on the Hearsay-II architecture [LESS77], a number of limitations in the architecture's data-directed, hypothesize-and-test control mechanism were detailed. Specifically, asking for something to be done (goal-directed control, such as precondition-action backchaining and subgoaling) is not easily accomplished in terms of the creation or modification of hypotheses. Also, there are no appropriate control data structures for relating in a detailed way the cooperative and competitive relationships among knowledge-source (KS) instantiations. The approach to scheduling taken in the Hearsay-II speech understanding system has a statistical and instantaneous character; there is no explicit planning of sequences of KS executions.

These limitations have become increasingly apparent to us and others as the architecture has been applied to different task domains. In our work on distributed interpretation (in which each element in a network of semi-autonomous Hearsay-II systems interprets a different part of the environment), we have found it difficult to coordinate in a globally coherent way the problem-solving of individual systems based solely on the use of data-directed control [LESS80], [LESS81a] [1].

Likewise, Nii and Feigenbaum with SU/X [NII78], Engelmore and Nii with SU/P [ENGE77], and Balzer, et al., with Hearsay-III [BALZ80] recognized these limitations and consequently enhanced the control capabilites of the Hearsay-II architecture. These enhancements permit more control over scheduling by allowing the KS scheduling queues to be manipulated under program control. However, these modifications do not explicitly formalize the relationship between goal- and data-directed control nor the relationship among KS instantiations [2]. Such relationships are left to the user to build. We feel these relationships are the key to implementing sophisticated control regimes in the multi-level and cooperating KS model of problem-solving posited by the Hearsay-II architecture.

In this paper, we first review the data-directed scheduling mechanisms of Hearsay-II and the assumptions behind this approach to control. Next, we indicate how data-directed and goal-directed control can be integrated into a single, uniform framework for control through the generation of goals from data-directed events.

---

[1] Nilsson in a recent workshop on Distributed AI (DAI) [DAVI80a] has made the case that one of the contributions of work in DAI is to make apparent the limitations of existing centralized problem-solving techniques.

[2] There has been work by Nilsson [NILS79] and de Kleer, et al., [deKL79] on this problem. Their approaches are, however, quite different in detail to the approach suggested here due to the strong differences in the underlying problem-solving model that each researcher has used.

Finally, we discuss some of the conceptual and implementation issues that need to be solved within this integrated framework for control.

2.0 Data-Directed Hearsay-II Scheduling

Figure 1 presents a high-level schematic for data-directed control in Hearsay-II. The action of executing a knowledge-source (KS) may result in changes to the blackboard (BB), such as adding new hypotheses, modifying the ratings of hypotheses, or changing the structural relationships among hypotheses. Associated with each type of data-directed event is a set of KSs that potentially can be executed based on the occurrence of that event type at a particular level in the BB. The BB event table contains the mapping between event types/levels and KSs. For each interested KS, a KS precondition procedure is executed to determine whether there is sufficient information on the BB for the KS to be executed. If so, a KS instantiation is created and placed on the scheduling queue.

The KS instantiation includes a description of the KS's stimulus and response frame. The stimulus frame specifies the stimulus hypotheses responsible for the creation of the KS instantiation. The response frame is an abstract description of the type of the BB events (type of BB modification, its BB level and region, and expected belief value of output hypotheses) that can result when the KS is executed. The scheduler uses this stimulus/response frame characterization of the KS instantiation, together with global state measures in the focus-of-control database, to calculate a priority rating for executing the KS. After a KS is executed, the KS instance with the highest priority rating is executed next.

This approach to scheduling can be characterized as instantaneous -- only the immediate effects on the state of problem-solving, as specified in the response frame, are considered. There is no inference process used to determine the effects of executing a KS beyond its immediate effects on the system state. The scheduler uses a single-shot evaluation function for determining the KS instance rating (see Hayes-Roth & Lesser [HAYE77] for details) [3].

Another aspect of this instantaneous approach to scheduling occurs when there is insufficient information for the KS precondition procedure to instantiate the KS. In this situation, the scheduler makes no record of which information is missing, and therefore has no way of re-evaluating its priority calculation so a KS that can generate the missing information will be rated higher or instantiated if already not present. (The scheduler does not use any precondition-action backchaining relationships

---

[3] The KS instance rating is re-evaluated if its stimulus hypotheses are modified or the relationship between the response frame and global state measure is changed.

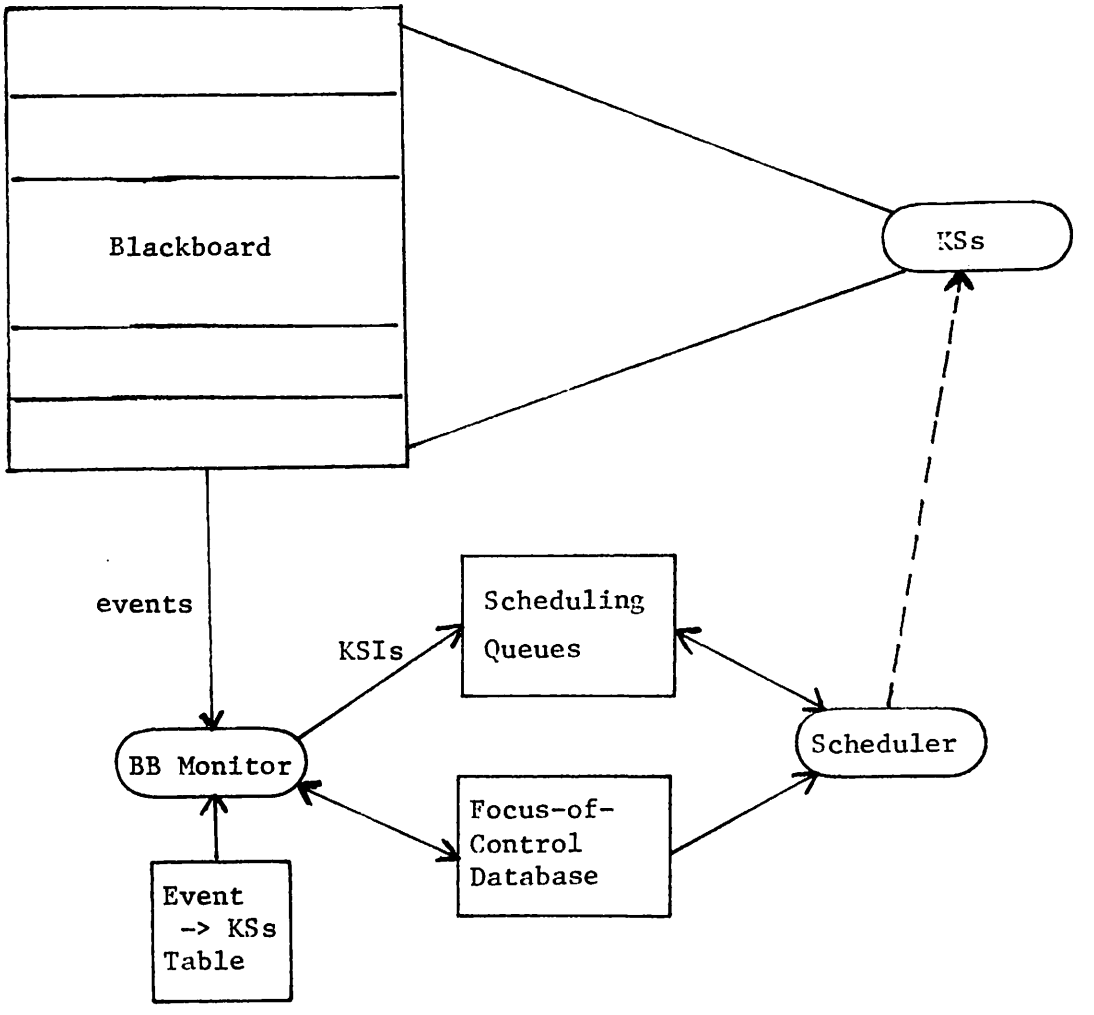


Figure 1: Data-Directed Hearsay-II Architecture

among KS instantiations.) Rather, it is assumed that if the information is really important it will eventually be generated based on normal scheduling considerations.

This instantaneous view of scheduling, which does not use detailed measures of the cooperative and competitive relationships among KS instantiations, is based on the following assumptions. The most fundamental assumption is that the errorfulness and incompleteness of the input data and KS processing make it impossible to accurately determine the consequences of executing a sequence of KSs. Thus, the control of processing must be constantly adapted as data is worked on and new information is generated. This suggests that a statistical and instantaneous approach to scheduling is the most pragmatic approach and that consideration of the details of future KS processing would be wasted effort.

There is also a set of more subtle assumptions that justified this approach to scheduling in the Hearsay-II speech understanding system. First, the speech domain did not provide strong high-level expectations about the types and locality of intermediate hypotheses. Additionally, an attempt to perform the action of subgoaling through the top-down elaboration of hypotheses in the speech domain would have resulted in a combinatorial explosion of hypotheses. Thus, it was unproductive to either bias the scheduling of KS activity or restrict KS processing based on hypotheses elaborated from high-level expectations. The only expectation-driven KS activity occurred when predicted hypotheses were verified. However, this activity was based only on the predictions derived from hypotheses generated bottom-up from the data, and the verifying KS had all the information necessary to perform its calculation. Therefore, there was no use for any scheduling mechanism based on subgoaling or precondition-action backchaining.

A second assumption was that the control of redundant KS processing due to alternative ways of deriving a hypothesis and the application of highly specialized KSs (e.g., a KS that can disambiquate between two highly-rated competing hypotheses such as "sit" and "split") could be handled through the normal data-directed scheduling mechanisms. It was felt that this could be accomplished by prioritizing different types of KS activity and by having complex KS precondition rules. Therefore, a detailed analysis of the history of KS activity and the state of processing was not required by the scheduler.

These assumptions allowed the use of an instantaneous, statistical scheduler in the Hearsay-II speech-understanding system. However, these assumptions are not valid in many task domains in which a Hearsay-II style problem-solving architecture is appropriate. In fact, the performance problems with the initial CO configuration of KSs in the Hearsay-II speech system (see Lesser and Erman [LESS77]) stemmed in part from KS interaction patterns which violated some of these assumptions. In hindsight, the scheduler for the CO configuration should have implemented some form of precondition-action backchaining and

subgoaling to permit more detailed control of KS activity.

To remedy these control problems within the basic Hearsay-II architecture, we next present an augmented version of the architecture that integrates data- and goal-directed control of KS activity via the generation of goals from blackboard events. Within this augmented architecture, a wide range of scheduling paradigms can be implemented efficiently: from those based on an instantaneous, statistical and data-directed approach to those based on complex planning of goal-directed activity. In this way, the system developer can tailor the control to the specifics of the task domain and KS configuration.

3.0 Goal-Directed Hearsay-II Scheduling

Figure 2 presents a high-level schemantic of Hearsay-II as modified to accomodate goal-directed scheduling. A second blackboard, the goal BB, is added that mirrors the original (data) BB in dimensionality. The goal BB contains goals, each representing a request to create a particular state of hypotheses on the data BB in the (corresponding) area covered by the goal. For example, a simple goal would be a request for the creation of a hypothesis above a given belief in a specified area of the data BB.

The integration of data-directed and goal-directed control into a single framework is based on the following observation:

> The stimulation of a KS precondition process in the data-directed architecture not only indicates that it may be possible to execute the KS, but that it may be desirable to do so in order to achieve the goal implicit in the KS response frame.

In order to make these implicit goals explicit, we split the event -> KSs mapping into two steps: event -> goals and goals -> KSs. The BB monitor watches for the occurrence of a data BB event, but instead of placing KS instantiations on the scheduling queue (if the KS preconditions are satisfied), it uses the event -> goals mapping to determine the appropriate goals to generate from the event and inserts them onto the goal BB. These goals represent the implicit goals contained in the event -> KSs mapping used by the original BB monitor.

Goals may also be placed on the goal BB from external sources. In a centralized system, it may be desirable to place a high-level goal onto the goal BB to bias the system towards developing a solution in a particular way. In our work on distributed interpretation [LESS80], [LESS81a], the exchange of goals between goal-directed Hearsay-II systems provides an effective means of coordination.

A new control component, the planner, is also added to the architecture. The planner responds to the insertion of goals on the goal BB by developing plans for their achievement. The

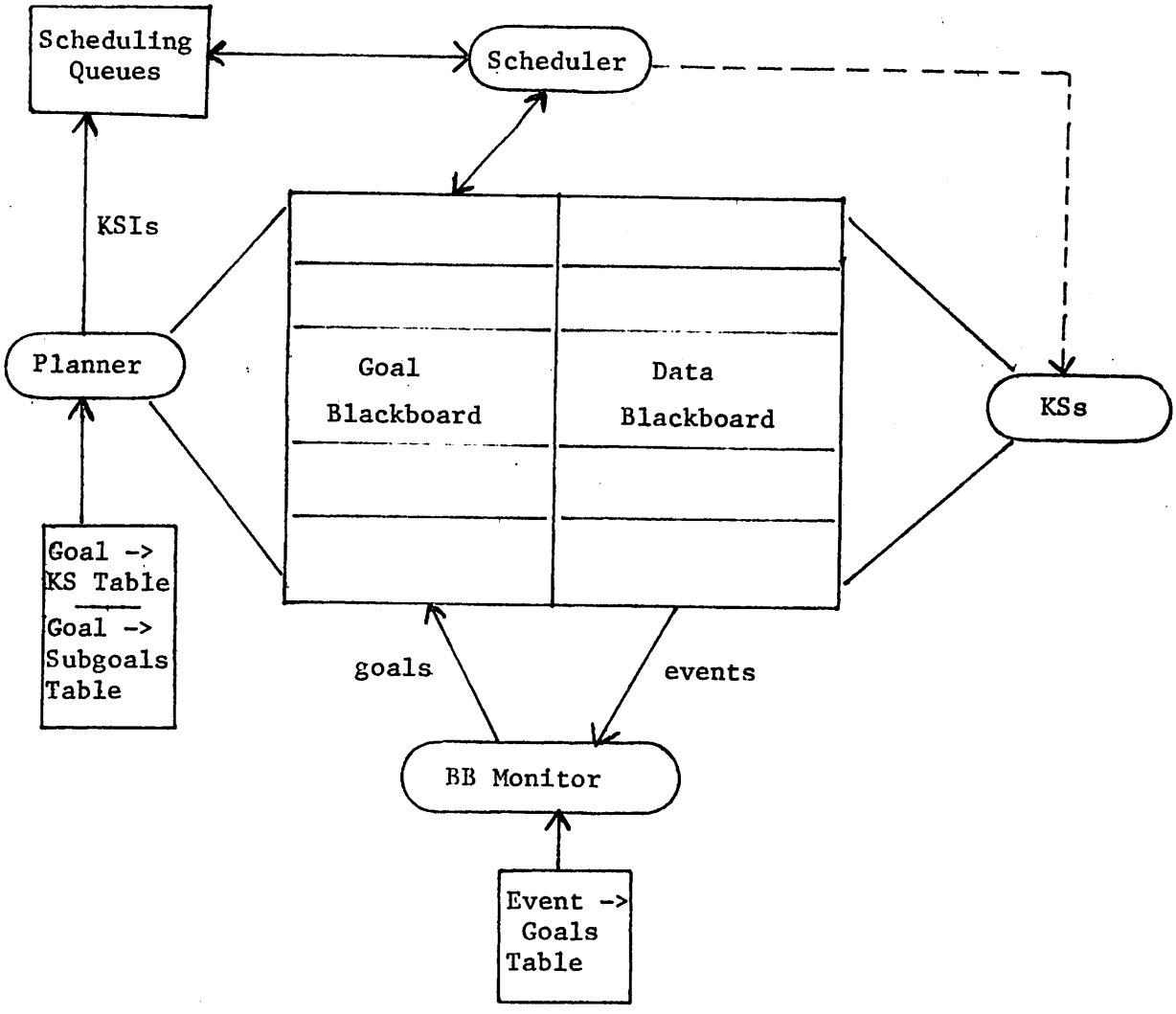


Figure 2: Goal-Directed Hearsay-II Architecture

goal -> KSs mapping is used by the planner to instantiate one or more KSs which can potentially satisfy the goals. The scheduler then uses the relationships between the KS instantiations and the goals on the goal BB as a basis for its scheduling decisions. In this way we have made explicit the relationship among BB events, goals, and KS instantiations, and thus can treat data-directed and goal-directed control within the same framework.

Figure 3 illustrates how a simple bottom-up synthesis sequence looks in the goal-directed architecture. For simplicity, we assume that each event triggers only one goal and only one KS can satisfy each goal. Hypothesis HA is inserted onto the data BB on level LDA (Figure 3a). That event causes the BB monitor to create goal GB onto the goal BB level LGB (Figure 3b). GB is a request to create a hypothesis supported by HA on LDB. The planner then looks at GB and instantiates KS KSIB with HA as its stimulus and places it onto the scheduling queue (Figure 3c). When KSIB executes, it creates hypothesis HB. The BB monitor responds to this event and creates goal GC (Figure 3d). It also checks to see if hypothesis HB can help to satisfy any goals on the goal BB. In this case, HB completely satisfies goal GB (Figure 3e). The planner next instantiates KSIC to achieve GC and places it onto the scheduling queue (Figure 3f). When KSIC executes it creates HC and the BB monitor marks GC as satisfied (Figure 3g).

In the above example, all necessary lower-level hypotheses exist when each KS is instantiated. Instead, suppose the goal GC is inserted first before any hypotheses on LDA (Figure 4a). Given only synthesis KSs, a data-directed system must wait until low-level hypotheses are inserted on LDA before instantiating KSs. On the other hand, the planner in the goal-directed system can immediately go ahead and instantiate KSIC (Figure 4b). The planner is provided with an inverse operator model (action -> precondition) of each KS, which it uses to determine that (precondition) goal GB must be achieved before KSIC can be executed (Figure 4c). To achieve GB, the planner instantiates KSIB which requires GA to be satisfied before it can execute (Figure 4d) [4]. When HA is eventually inserted (Figure 4e), the goal/KS instantiation structure is used by the scheduler to execute KSIB and KSIC as a multiple-KS plan to finally achieve GC (Figures 4f,g).

The planner is also provided with domain knowledge in the form of a goal -> subgoal mapping for decomposing high-level goals into lower-level ones. This allows it to determine GB directly from GC and GA from GB (Figures 5a,b). In this case, KSIB and KSIC are not instantiated by the planner until their respective goals (GB and GC) have their subgoals satisfied (Figures 5c-g). In fact, the intermediate goal GB can be omitted; the planner can directly determine the lowest level subgoals of the high-level

---

[4] If level LDA hypotheses are produced by a controllable sensor, GA can be used to request the insertion of hypotheses which satisfy GA.

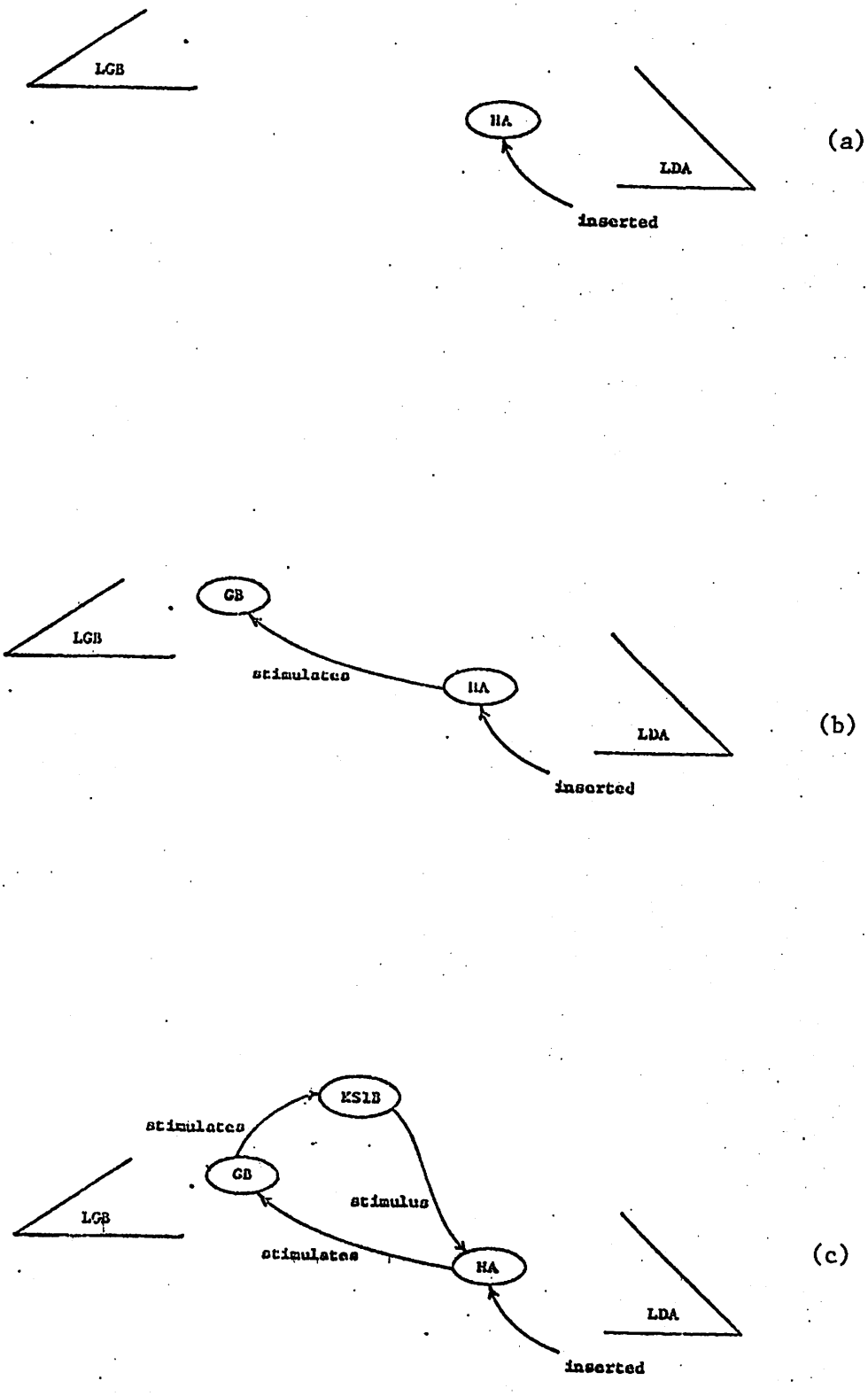
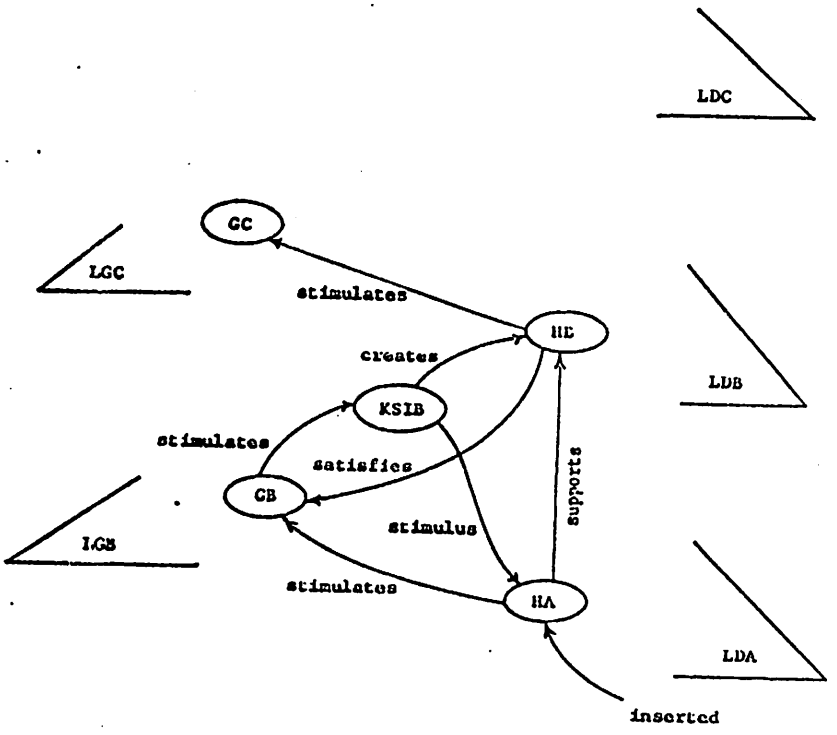
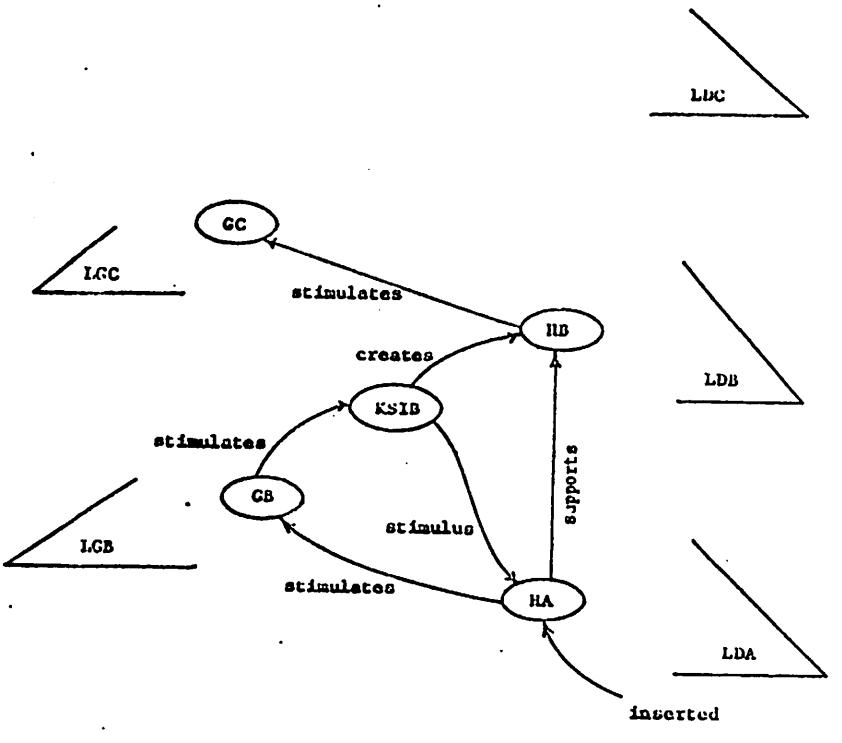


Figure 3: Bottom-Up Goal-Directed Processing

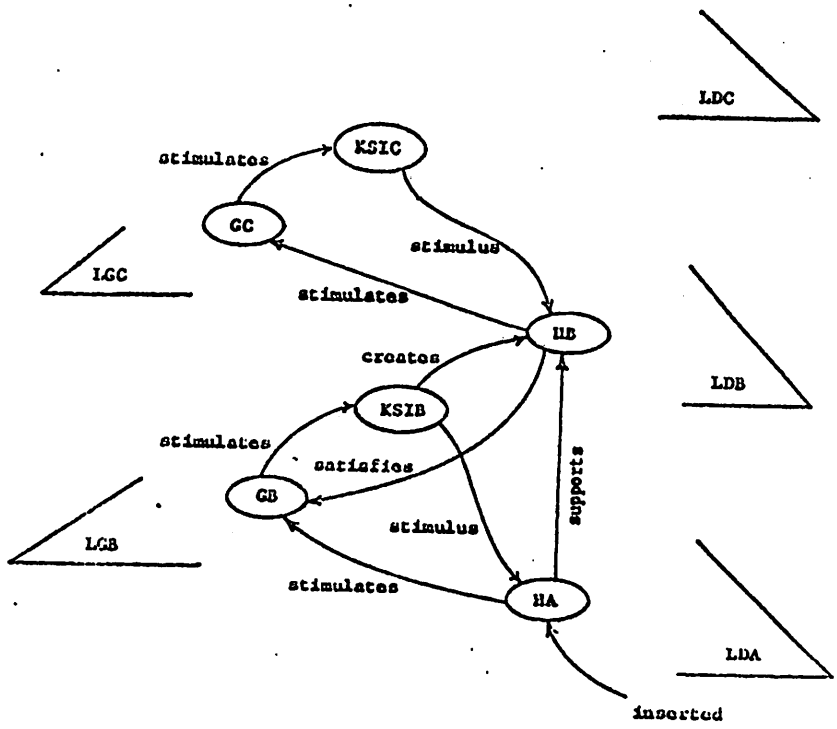


(d)

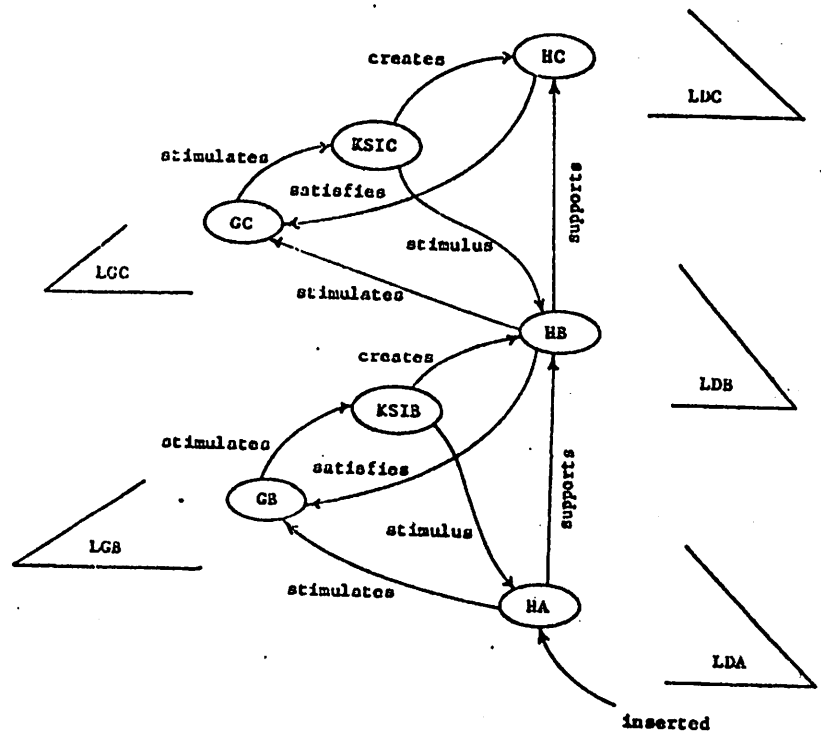


(e)

Figure 3: continued



(f)



(g)

Figure 3: continued

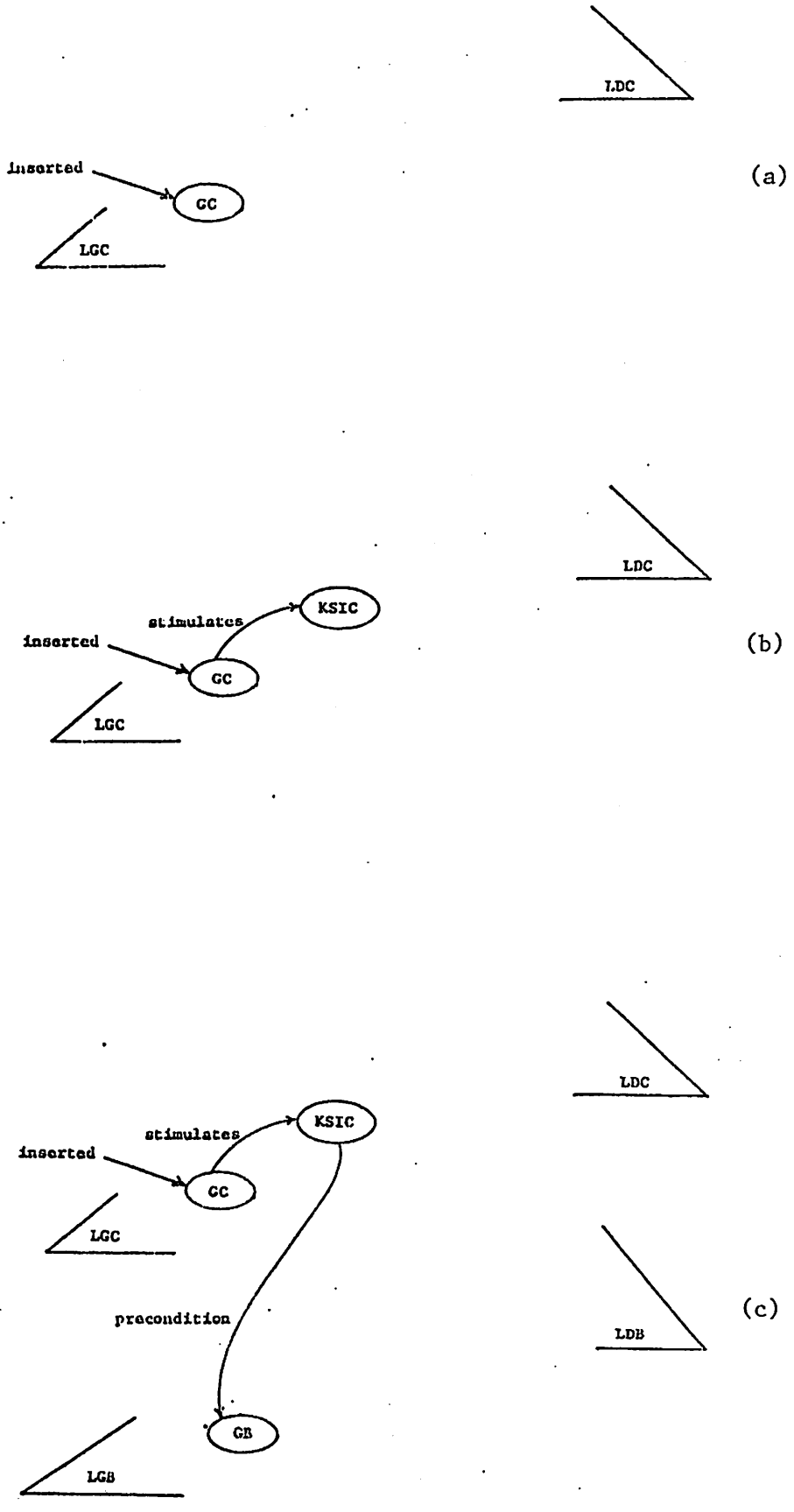
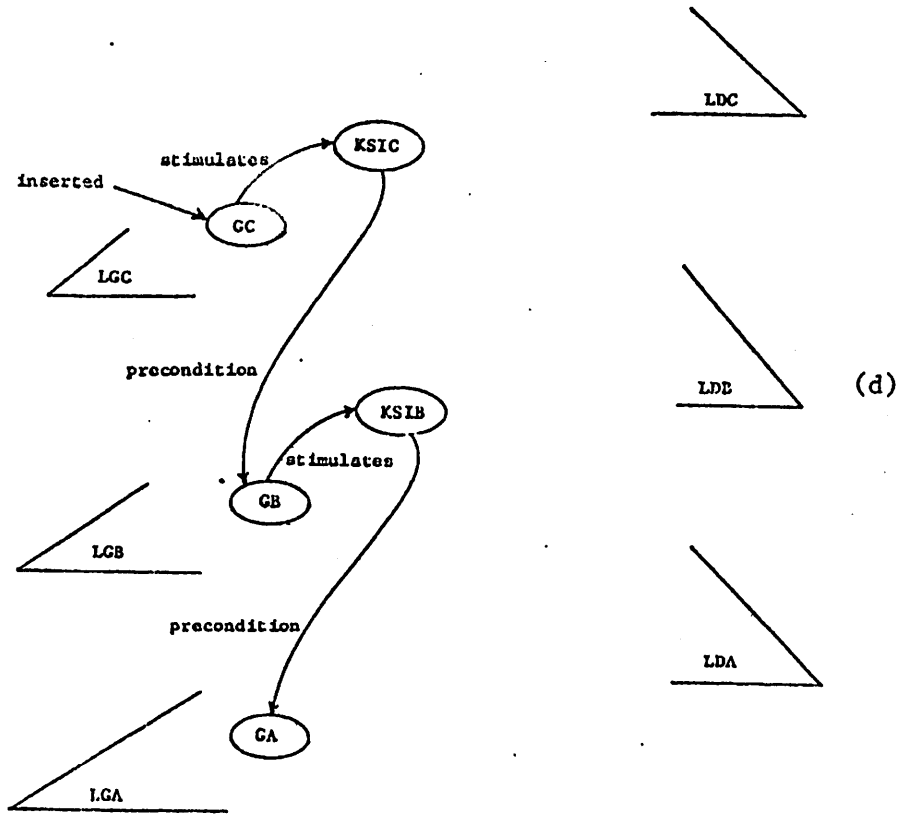
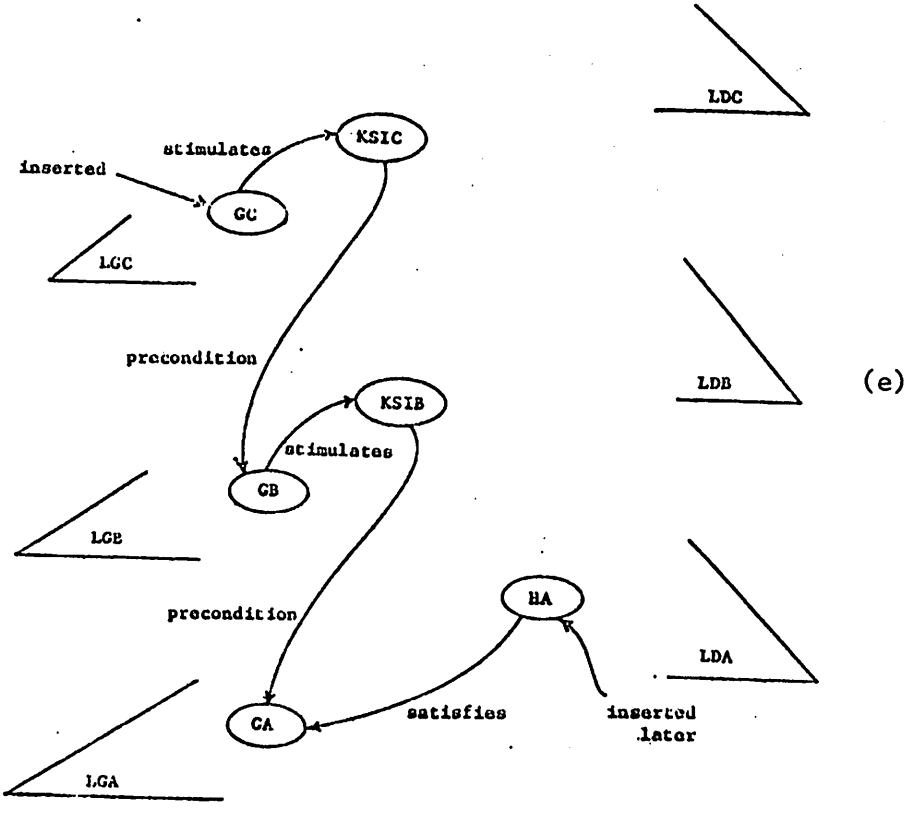


Figure 4: Precondition-Action Backward Chaining

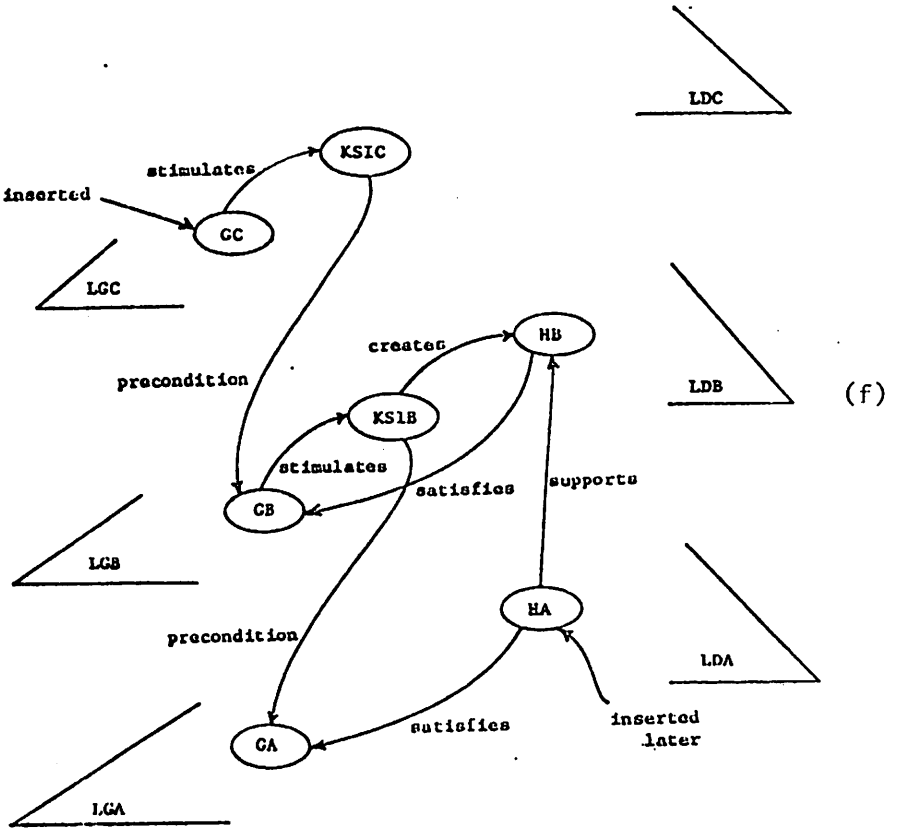


(d)

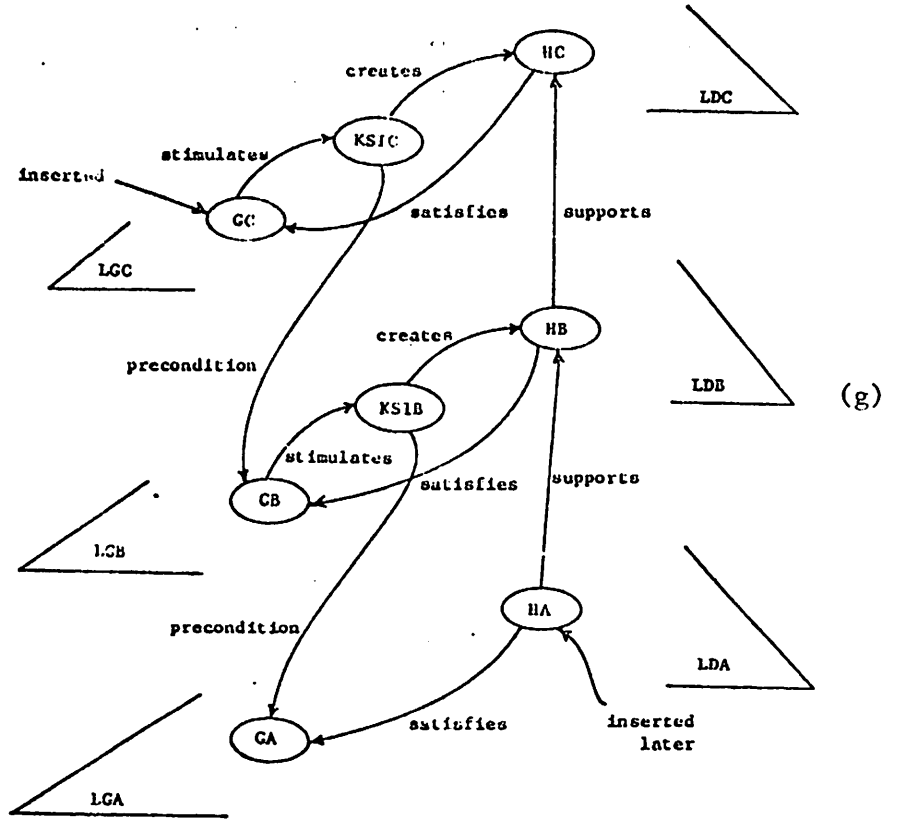


(e)

Figure 4: continued



(f)



(g)

Figure 4: continued

goal. Once HA is satisfied, the intervening goals can be created bottom-up, using the high-level to low-level subgoal relationship as a guide. In a multilevel system, this "level-hopping" can significantly reduce the overhead required to determine which low-level hypotheses can be driven up to satisfy a particular high-level goal (see Section 4).

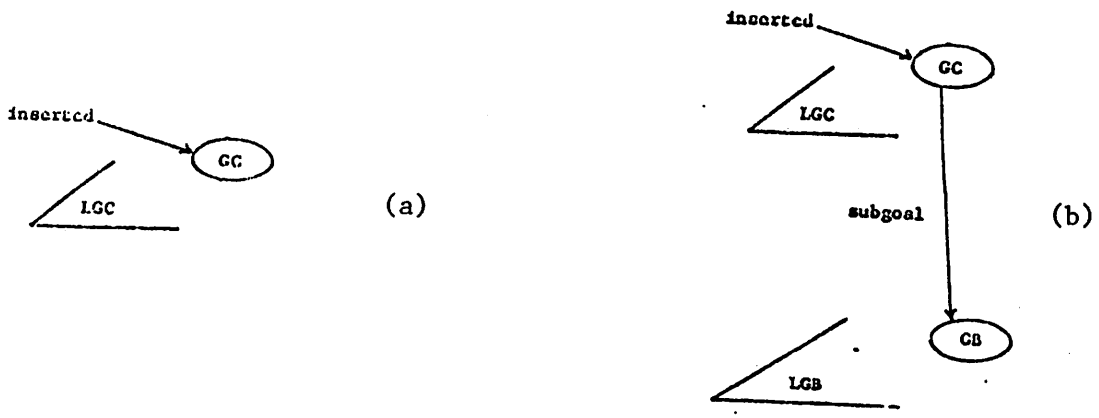
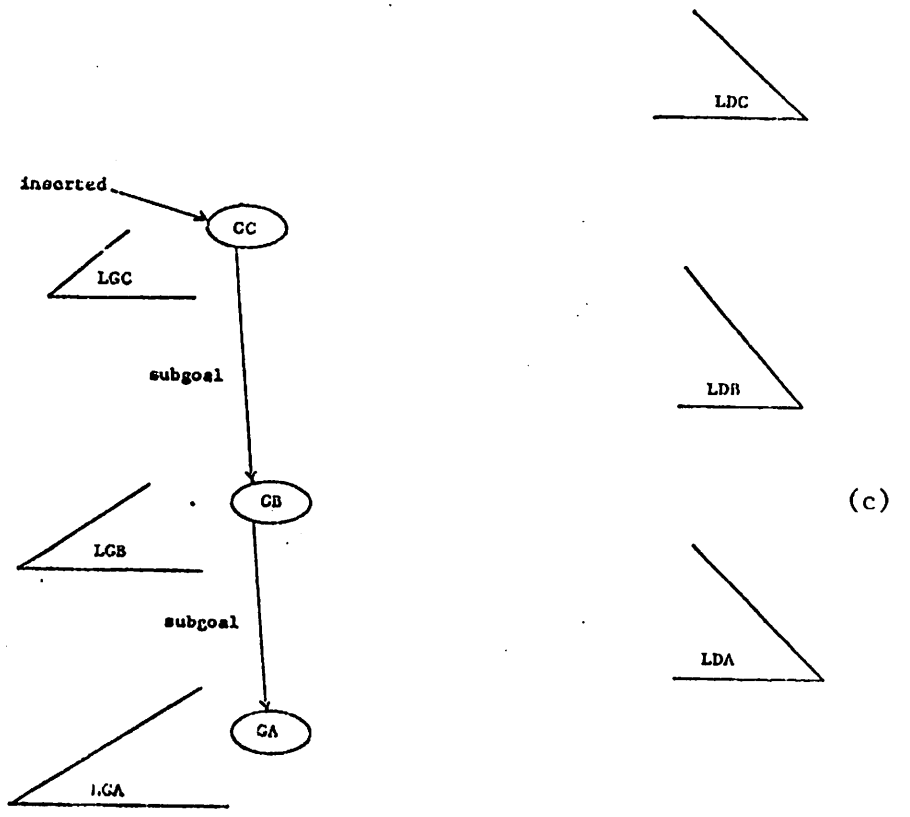


Figure 5: Subgoaling

The goal-directed approach permits all three of these goal-processing activities to be performed in an opportunistic way by the planner. Highly rated low-level hypotheses can be driven up in a data-directed fashion while high-level goals generated from strong expectations can be subgoaled downward to control low-level synthesis activities. Similarly, processing in low rated areas can be stimulated if a highly rated KS requires the creation of a precondition goal in that area.

Control decisions in the goal-directed architecture can be made or deferred at a number of points in the data BB event to KS execution process, based on the availablity and reliability of control information. When a goal is created by the BB monitor, it is assigned an initial rating of the goal's importance that is based on the belief of its stimulus hypotheses, the rating of any KS instances which require satisfaction of the goal as a precondition, the rating of any goals which have the goal as a subgoal, and on any externally supplied priority (for externally inserted goals). These ratings are used to direct planning activity to higher rated goals. The BB monitor also serves as a "goal filter" by placing only those goals with a priority rating above a specified goal-threshold onto the goal BB.

Similarly, the planner can use a second dynamic threshold value to ignore goals placed onto the goal BB which are rated below its threshold. The distinction between the two is that the planner can eventually reconsider goals on the goal BB which are currently below the planning-threshold, but goals which are not



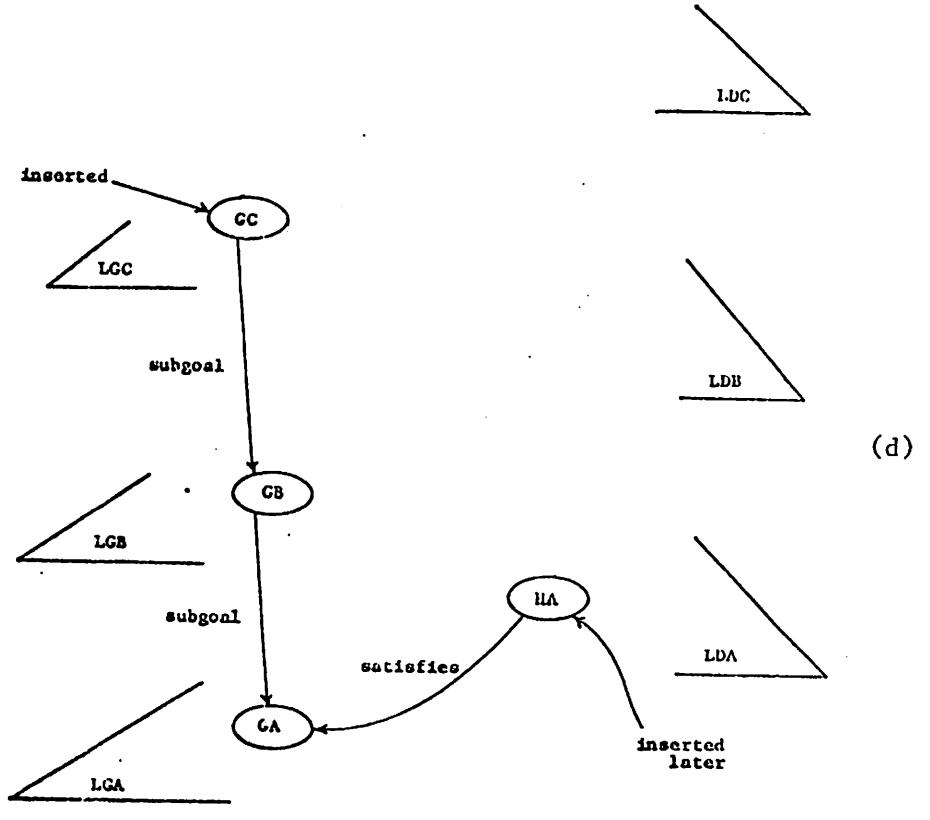
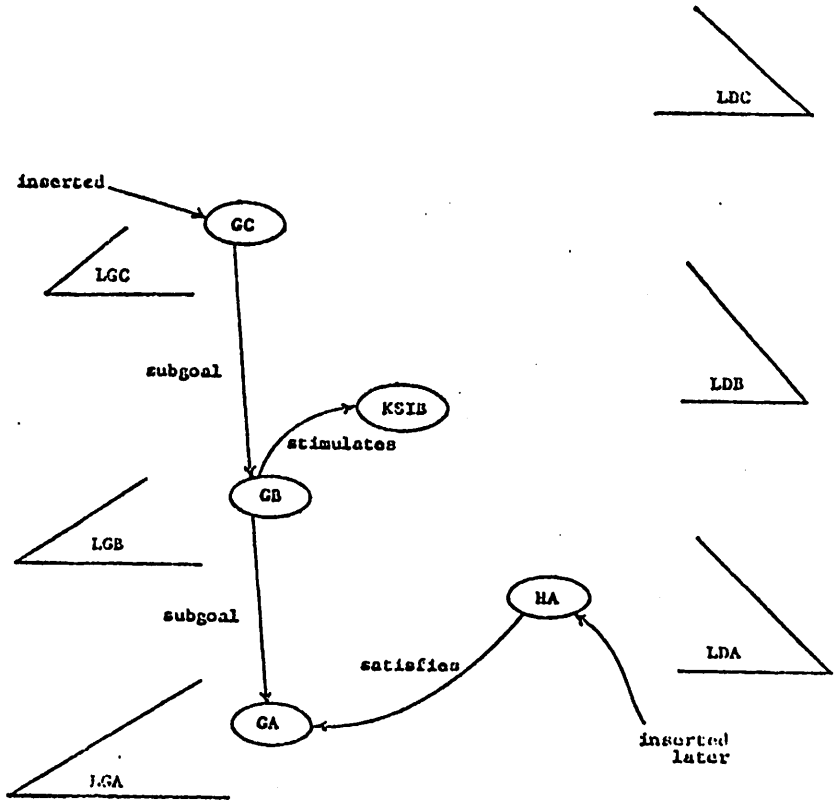
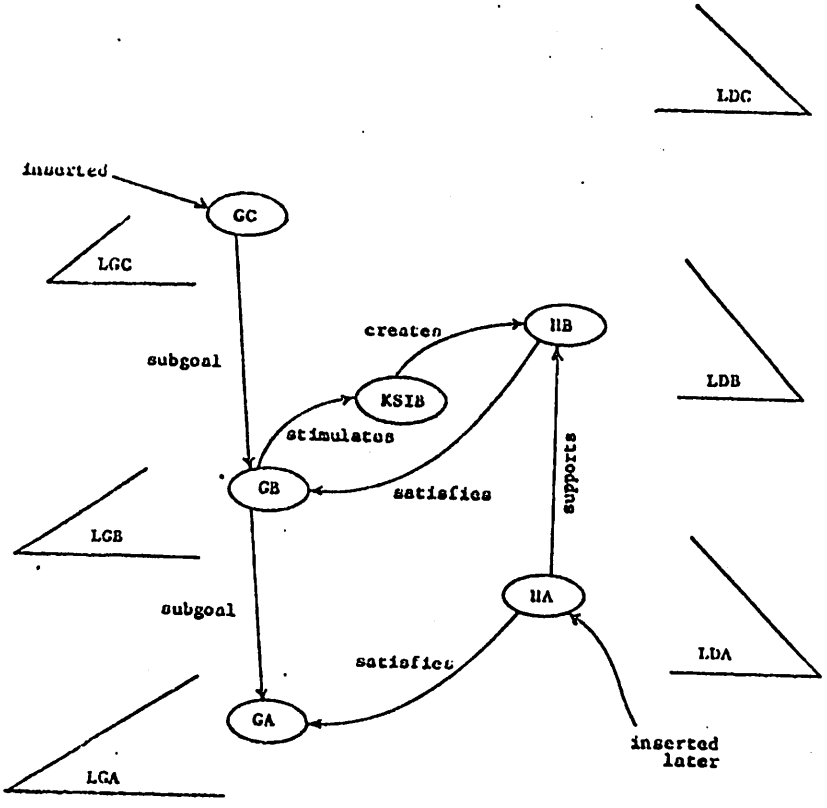


Figure 5: continued

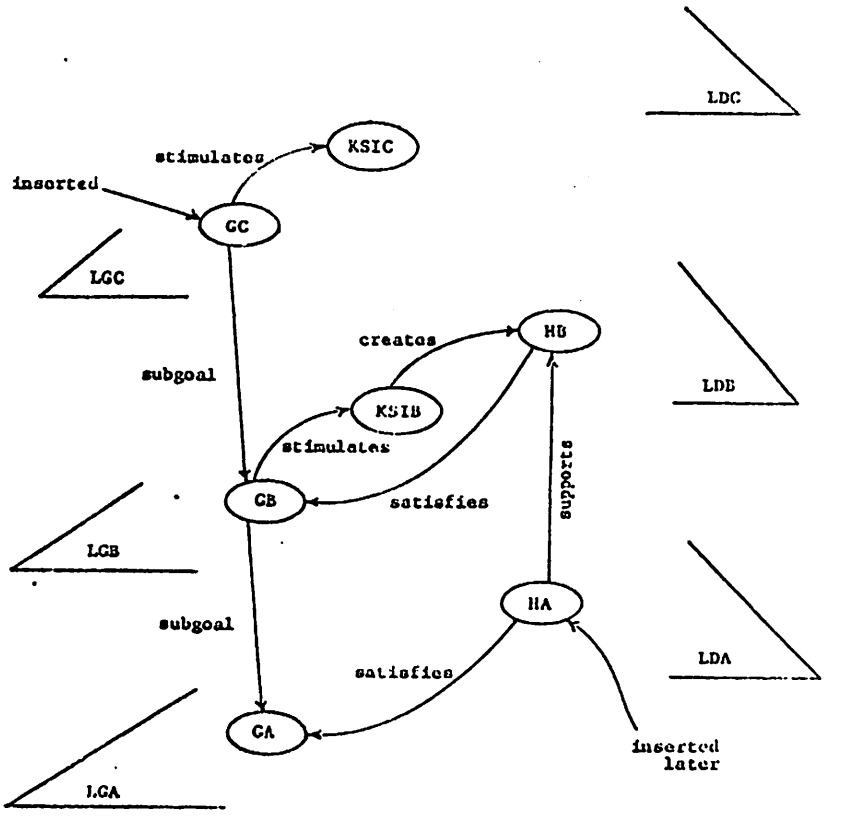


(e)

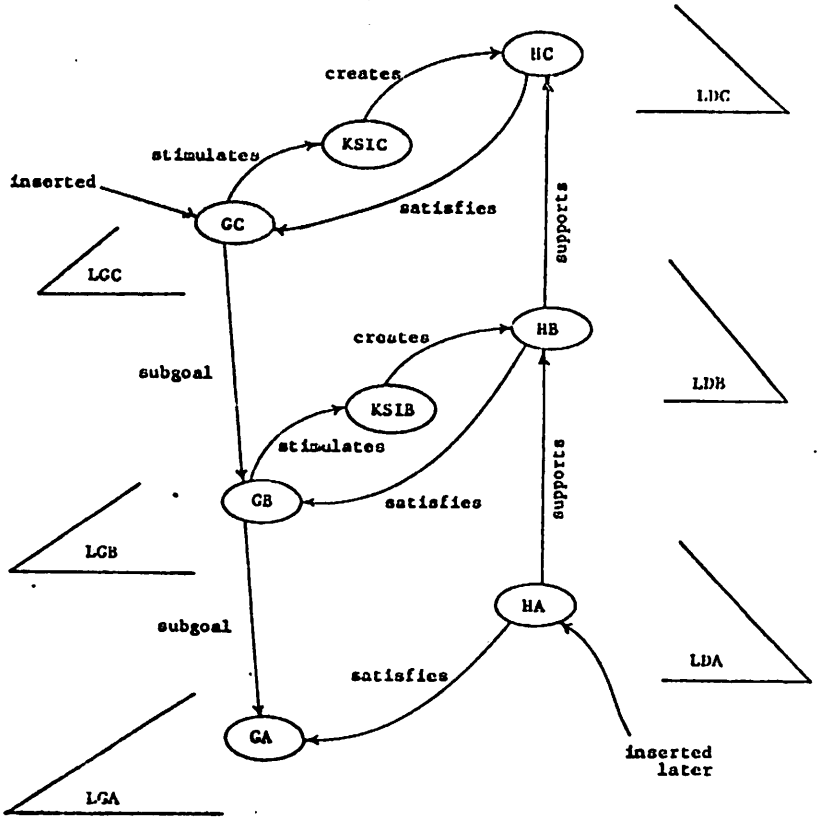


(f)

Figure 5: continued



(g)



(h)

Figure 5: continued

inserted by the BB monitor are lost and must be recreated by the planner from other planning activities.

The goal-directed architecture also provides the scheduler with additional information for selecting which competing KS instantiations to execute. The comparison among alternative KSs which achieve the same goal is explicitly available. Figure 6 shows two highly rated goals, each of which can be satisfied by a moderately expensive KS or a more expensive KS. However, the more expensive KS can satisfy both goals. Due to the lack of this information in the data-directed system, the single-shot scheduler would select the two moderately expensive KSs rather than the shared KS which satisfies both goals for a lower total cost.

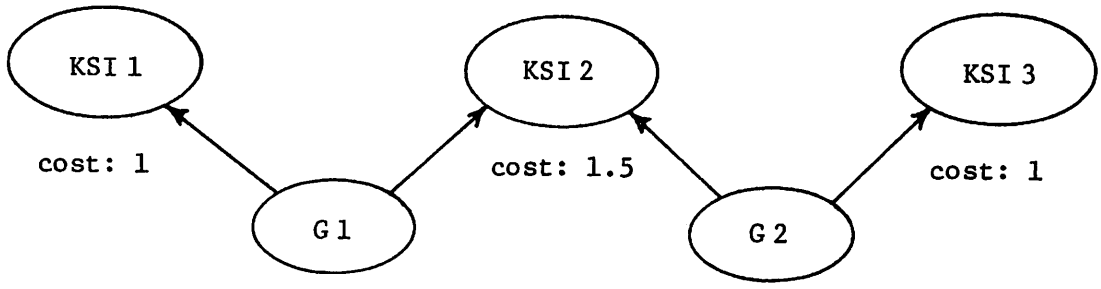


Figure 6: Choosing between Alternative KS Instantiations

The explicit goal structure also allows the planner to construct alternative strategies consisting of multi-step sequences of KS execution and goal satisfaction. The competitive and cooperative relationships between KSs and goals developed by the planner are then made available to the scheduler, which can use these relationships to forgo the execution of other KSs in a strategy which has failed, to eliminate redundant KS processing, and to effectively control the application of highly specialized KSs.

Two generalizations in the model of a KS are also important in the goal-directed approach. In the data-directed system, a KS uses its stimulus hypotheses as an input context for the generation of its output. In the goal-directed system, the planner can elect to supply a KS with only stimulus hypotheses, only goals, or both stimulus hypotheses and goals. In the first case, the KS functions as in the data-directed system. In the second case, the KS itself determines the input context (by searching for appropriate input hypotheses on the data BB) in order to best achieve the supplied goals. In the third case, the stimulus hypotheses are used as the input context (avoiding the data BB search) and the goals are used as an output filter (hypotheses which are outside the scope of the goals are not

created by the KS).

The second generalization of the KS model is the establishment of bi-directional communication between each KS and the planner. While the KS precondition process is used by the planner to determine if the major conditions necessary for the KS to achieve the goal are present, a KS may still fail due to the lack of "secondary" preconditions or to detailed incompatabilities among its input hypotheses. The KS reports the nature of the problem back to the planner, which can elect to create highly specific precondition goals for the KS or to choose another KS that can satisfy the goal in a different manner.

The unification of data- and goal-directed control provides the flexibility to implement a wide range of control mechanisms and eliminates a number of problems caused by a statistical and instantaneous approach to scheduling. However, the introduction of goal processing into the architecture also raises a number of conceptual and implementational issues. In the next section we discuss in more detail the representation of goals, the use of goals to control combinatorics, the details of goal processing, and the scheduling of goal processing activity versus KS activity.

4.0 Issues

A number of attributes are associated with each goal. The satisfaction specification attribute is a declaration of the desired state of hypotheses on the data BB that the goal represents. It serves as the basic means of communication between the BB monitor, the planner, KS precondition processes, and KSs. A second attribute associated with a goal is the minimum satisfaction specification, the minimum conditions under which the goal is considered satisfied. The satisfaction specification states what conditions are desired, and the minimum satisfaction specification indicates when the goal is sufficiently satisfied to allow processing to proceed to KS instantiations that require minimal achievement of the goal.

Another attribute is the rating of the goal's importance, which is used to direct the planner and to influence the rating of KS instantiations. In some domains the importance of goals may sharply decline due to time constraints on the resulting hypotheses. In this case it is useful to associate a "time-out" condition with the rating at which time it is recalculated. An estimate of the cost of achieving the goal is another useful goal attribute. Cost estimates can be used by the planner to choose between alternative strategies based upon the resources which must be expended in each. Similarly, an estimate of the probability of satisfying the goal can be used by the planner to choose a strategy which has the greatest chance of success.

Goals also contain a number of attributes linking them to other goals, KS instantiations, and hypotheses. These link attributes include: the goal's subgoals, goals which include it as a subgoal (supergoals), goals which are more abstract, goals

which are more specialized, the hypotheses which stimulated the creation of the goal, the hypotheses which satisfy the goal, and KS instantiations which require the goal to be achieved as a precondition.

Goals can be used to request a number of different types of KS processing based on the form of their satisfaction specification attributes. Specific-hypothesis goals request a change to be made to a particular (named) hypothesis. "Increase the belief of hypothesis HA by at least 20 percent" is a specific-hypothesis goal. Generic-hypothesis goals request the creation or modification of a single (unnamed) hypothesis which matches a set of specified attributes. "Create a hypothesis on level LBC at time 3 with belief greater than 0.5" is a generic-hypothesis goal. Area goals request the establishment of relationships among hypotheses in a specified area of the data BB. "Create at least 5 hypotheses with belief greater than HB on level LBC at time 3" is an example of an area goal.

Goals can also be characterized according to their duration. Single-shot goals remain active (the planner attempts their satisfaction) only until they are first satisfied. Single-shot goals may be restimulated by the BB monitor or by the insertion of external goals, but are distinct from continuous goals which always remain active. "Every time a hypothesis is created on level LDC that has a belief greater than 0.9, send it to node 2" is an example of a continuous goal.

Goals such as "create from stimulus hypothesis HA only those hypotheses with belief greater than 0.6" naturally implement threshold control of KS activity. Threshold control reduces the number of hypothesis inserted onto the data BB by treating KSs as generator functions which create only their highest rated output hypotheses and can be reinvoked later if lower rated hypotheses are desired.

As mentioned earlier, goal processing can also reduce the combinatorics associated with the top-down elaboration of hypotheses. Top-down elaboration is generally used for two different activities: the generation of the lower-level structure of a hypothesis (to discover details) and the determination of which existing low-level hypotheses should be driven-up to verify a high level hypothesis based on expectations (for focusing). Top-down elaboration of hypotheses is best suited only to the first activity -- subgoaling on the goal BB is a more effective way to perform expectation-based focusing. When hypothesis elaboration is used as a focusing technique, the elaboration process has to be conservative in order to reduce the number of hypotheses generated and to reduce the possibility of generated low-level hypotheses being used as "real data" by KSs in other contexts. Because subgoals are distinct from hypotheses, they can be liberally abstracted (such as supplying a range of values for an attribute) and underspecified (such as supplying a "don't care" attribute). Therefore, subgoaling the high-level goal of generating the expectation-based hypothesis (including the use of "level-hopping") avoids the combinatorial and context confusion problems associated with the use of top-down hypothesis

elaboration for focusing.

In addition to the use of data BB events to create new goals, these events must also be checked to determine if they can satisfy existing goals on the blackboard. This checking can be performed by the BB monitor, the planner, or a combination of both, depending on characteristics of the task domain. If the checking requires only simple, syntactic matching of the attributes of hypotheses with those of goals, then it should be implemented as part of the BB monitor. This is the approach we have taken in our implementation. However, if extensive task domain knowledge and processing resources are required for matching, then the planning module is the appropriate place to perform the checking. The philosophy is to keep the BB monitor a simple table-driven procedure with low processing overhead that is not scheduled, but executed as required. A combined approach, in which a quick syntactic check is first performed by the BB monitor, can also be used. If this check results in only partial matches, then the event and partially matched goals are passed on to the planner for more extensive analysis. In all of these cases, the parallel structure of the data and goal BBs facilitates the effective implementation of event/goal matching.

Goal merging, which involves recognizing that two goals can be satisfied by the same conditions, can be similarly implemented in the BB monitor, the planner, or a combination of both. Again the choice is based on the complexity of the operation required by the task domain.

Additional goal-processing operations also need to be implemented. These operations involve checking newly created goals for their relationships with existing goals, including the relationships of: goal/subgoal, goal/precondition-goal, goal/abstract-goal, and goal/specialized-goal. The need for determining these relationships depends on the task domain and planning strategies used in the system and, due to their complexity, seem best implemented in the planning module.

In order to perform complex goal processing, the planning module requires a number of basic operations on the goal BB. These operations are analogous to the basic operations provided for KS manipulations on the data BB and include: inserting goals onto the goal BB, accessing and modifying goal attributes, selective retrieval of goals based on their attributes, creating structural relationships between goals, and chaining through these structural relationships.

Complex goal-processing is not without cost, and as the overhead of goal processing increases, it is important to balance planning activities with KS execution. We feel the scheduler should perform the allocation of processing resources -- both to the planner and the KSs. The ratings and the relationships between the goals and KS instantiations provide the scheduler with the information necessary to determine the best course for improving the state of the system. Techniques for reasoning about the balance between planning the consequences of actions versus performing them to discover the result are needed. The work by

Feldman and Sproull [FELD77] is a first step in this direction.

Complex goal-processing raises the issue of whether the planner itself should be implemented as a data-directed system with its own planning KSs and whether that system should be augmented with a (meta) goal BB and goal-processing mechanisms. Goals requesting changes on the goal BB can be used to explicitly represent the problem-solving strategies of the system [HAYE79], [DAVI80b], [STEF80]. Such meta-level goals would represent strategies for the planner. Subgoaling a high-level, expectation-based goal to low-level goals and then driving-up the appropriate low-level hypotheses upward is an example of a useful strategy which could be represented by a meta-level goal.

If there are a number of meta-level goals, a strategy for choosing between them is needed. This raises the problem of controlling the meta-controller, and so on. One approach is to add control layers until the highest level controller becomes a simple procedure. This is the approach we are currently pursuing. A second approach is to introduce a controller which can reason about its own control decisions as well as those it is making for the lower levels (while avoiding the problems associated with self-reference).

5.0 Conclusion and Future Research

We have shown how data- and goal-directed control can be naturally integrated into a single uniform control framework, permitting the development of a wide range of different scheduling and planning strategies for controlling knowledge source (KS) activity. This framework increases the number of task domains in which the multi-level, cooperative KS model of problem-solving (used in the Hearsay-II architecture) is an effective approach.

We are currently exploring this integrated control framework in the task domain of distributed interpretation as a means of obtaining globally cohesive behavior in a network of semi-autonomous goal-directed Hearsay-II systems [LESS81b]. A preliminary version of this system, with only a rudimentary planning module, has been implemented. We are currently testing the power of the integrated control framework by developing a sophisticated planning module for coordinating KS activity within a node with the problem-solving requirements of other nodes in the network. Experiments with this planning module should lead to an improved understanding of the appropriate balance between data- and goal-directed activity, the overhead associated with goal-processing activities, and a number of detailed issues regarding the mechanisms necessary for implementing complex goal processing.

Acknowledgment

Many of the issues relating to goal-processing evolved during discussions with Jasmina Pavlin and Larry Lefkowitz. Lee Erman and Dave McDonald provided helpful comments on early drafts of this paper. We also acknowledge the implementation efforts of Eva Hudlicka.

References

BALZ80] R. Balzer, L.D. Erman, P. London, and C. Williams. "Hearsay-III: A Domain-Independent Framework for Expert Systems." Proceedings First National Conference on Artificial Intelligence, pages 108-110 (August 1980).

DAVI80a] R. Davis. "Report on the Workshop on Distributed AI." SIGART Newsletter, No. 73, pages 42-52 (October 1980).

DAVI80b] R. Davis. "Meta-Rules: Reasoning About Control." AI Memo 576, Laboratory for Artificial Intelligence, Massachusetts Institute of Technology, Cambridge, Massachusetts (March 1980).

deKL79] J. de Kleer, J. Doyle, G.L. Steele, Jr., and G.J. Sussman. "Explicit Control of Reasoning." Artificial Intelligence: An MIT Perspective, Volume 1, P.H. Winston and R.H. Brown (Eds.), MIT Press, Cambridge, Massachusetts, pages 93-116 (1979).

ENGE77] R.S. Engelmore and H.P. Nii. "A Knowledge-Based System for the Interpretation of Protein X-ray Crystallographic Data." Technical Report Stan-CS-77-589, Computer Science Department, Stanford University, Stanford, California (1977).

FELD77] J.A. Feldman and R.F. Sproull. "Decision Theory and Artifical Intelligence II: The Hungry Monkey." Cognitive Science, Vol. 1, No. 2, pages 158-192 (April 1977).

HAYE77] F. Hayes-Roth and V.R. Lesser. "Focus of Attention in the Hearsay-II System." Proceedings Fifth International Joint Conference on Artificial Intelligence, pages 27-35 (August 1977).

HAYE79] B. Hayes-Roth and F. Hayes-Roth. "A Cognitive Model of Planning." Cognitive Science, Vol. 3, No. 4, pages 275-310 (October-December 1979).

LESS77] V.R. Lesser and L.D. Erman. "A Retrospective View of the Hearsay-II Architecture." Proceedings Fifth International Joint Conference on Artificial Intelligence, pages 790-800 (August 1977).

LESS80] V.R. Lesser and L.D. Erman. "Distributed Interpretation: A Model and Experiment." IEEE Transactions on Computers, Vol. C-29, No. 12, pages 1144-1163 (December 1980).

LESS81a] V.R. Lesser and D.D. Corkill. "Functionally Accurate, Cooperative Distributed Systems." IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-11, No. 1, pages 81-96 (January 1981).

LESS81b] V.R. Lesser, et al. "A High-Level Simulation Testbed for Cooperative Distributed Problem-Solving." Technical Report, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts (June 1981).

NII78] H.P. Nii and E.A. Feigenbaum. "Rule-Based Understanding of Signals." Pattern-Directed Inference Systems, D.A. Waterman and F. Hayes-Roth (Eds.), Academic Press, New York, pages 483-501 (1978).

NILS79] N.J. Nilsson. "A Production System for Automatic Deduction." Machine Intelligence 9: Machine Expertise and the Human Interface, J.E. Hayes, D. Michie, and L.I. Mikulich (Eds.), Chichester, pages 101-126 (1979).

STEF80] M.J. Stefik. "Planning With Contraints." Ph.D. thesis, Technical Report STAN-CS-80-784, Computer Science Department, Stanford University, Stanford, California (January 1980).