Report Number: WUCS-87-15

1987-01-01

Belief Maintenance Systems: Initial Prototype Specification

Roseanne M. Fulcomer, William E. Ball, and John P. Tadlock

A fundamental need in future information systems is an effective method of accurately representing and monitoring dynamic, real-world situations inside a computer. Information is represented using an Extended Open World Assumption (EOWA), in which the data are explicitly true or false. Reasoning within the EOWA is done through the use of a dynamic dependency net which only represents those beliefs and justifications that are both currently valid and in current use. In this paper, we present definitions and uses of the EOWA and dynamic dependency net in our current research of developing a database with which we can use... Read complete abstract on page 2.

Belief Maintenance Systems: Initial Prototype Specification

Roseanne M. Fulcomer, William E. Ball, and John P. Tadlock

Complete Abstract:

A fundamental need in future information systems is an effective method of accurately representing and monitoring dynamic, real-world situations inside a computer. Information is represented using an Extended Open World Assumption (EOWA), in which the data are explicitly true or false. Reasoning within the EOWA is done through the use of a dynamic dependency net which only represents those beliefs and justifications that are both currently valid and in current use. In this paper, we present definitions and uses of the EOWA and dynamic dependency net in our current research of developing a database with which we can use deductive reasoning with limited resources. A prototype has been implemented for determining the existing problems of creating such a belief management system for operation in real-world applications.

BELIEF MAINTENANCE SYSTEMS: INITIAL PROTOTYPE SPECIFICATION

Rosanne M. Fulcomer, William E. Ball
and John P. Tadlock

WUCS-87-15

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
Saint Louis, MO 63130-4899

This work was supported by McDonnell Douglas Company under contract Z71021.

Belief Maintenance Systems:

Initial Prototype Specification

Rosanne M. Fulcomer
William E. Ball
Washington University
Department of Computer Science
St. Louis, MO 63130

John P. Tadlock
McDonnell Aircraft Company
Artificial Intelligence Technology Group

Abstract

A fundamental need in future information systems is an effective method of accurately representing and monitoring dynamic, real-world situations inside a computer.

Information is represented using an Extended Open World Assumption (EOWA), in which the data are explicitly true or false. Reasoning within the EOWA is done through the use of a dynamic dependency net which only represents those beliefs and justifications that are both currently valid and in current use.

In this paper, we present definitions and uses of the EOWA and dynamic dependency net in our current research of developing a database with which we can use deductive reasoning with limited resources. A prototype has been implemented for determining the existing problems of creating such a belief management system for operation in real-world applications.

Acknowledgements: This work was supported by McDonnell Douglas Company under contract Z71021.

TABLE OF CONTENTS

Belief Maintenance Systems: Initial Prototype Specification

Rosanne M. Fulcomer
William E. Ball
John P. Tadlock

WUCS-87-15

1. Introduction

1.1. What is a Belief Management System?

A Belief Management System (BMS) is a system that contains a vast amount of data and rules governing that data to determine belief in deduced information. Its main capability is to efficiently and confidently make inferences and draw conclusions with minimal help from the user. It can deal with data and rule inconsistencies. Uses of such a system are many, but we have tested it on problems that range from animal recognition to airplane recognition.

1.2. Objectives

The objective of this paper is to present an overview of our approach to the problem of belief revision and maintenance. Current systems fall short in that they are domain specific and cannot be used in real-time applications. Our investigation includes the development of a system that will employ what we call a *cognizant database*. A cognizant database is a database used in conjunction with a belief maintenance operating under limited resources.

This paper describes an existing prototype system. The initial system developed can automatically draw inferences and conclusions, along with the ability to deal with incomplete information. The system's ability to draw inferences is embedded in an inference engine that uses beliefs in the database to determine which rules in the knowledge base will be satisfied. To deal with incomplete information, the system employs an assumption controller that questions the user to assert appropriate assumptions possibly leading to conclusions. This assumption controller, though primitive, will also deal with any contradictions occurring due to assumption making. The system can give reasons for its decisions or results.

1.3. Approach

One major difference in our approach to belief management as compared to the other approaches solving similar problems is the use of an Extended Open World Assumption. In general, the extension is in the use of assumption making which is not present in the previously

defined open world assumption [8].

Another difference is in the use of a dynamic dependency net, which does not remain the same as information changes. Storing dependencies and other information on beliefs dynamically will decrease the amount of contradiction occurring, and hence decrease the amount of consistency checking needed to maintain the dependency. Also, anything dynamic in nature is a closer representation of a real world model than a static representation. The information needed by the user will be computed directly and interactively if it is not able to be deduced using logical modus ponens.

2. Background

If we look at a database from the point of view of first-order logic we can create what is called a *deductive database* [3]. Deriving new facts may be considered a form of theorem proving. From this viewpoint there are at least three basic problems that must be considered and solved:

(1) The manner in which negative data are to be treated.

(2) The null value problem in which the value of a data item is missing.

(3) The indefinite data problem, in which one knows that "P(a) or P(b)" is true but not if "P(a)" is true, "P(b)" is true, or both are true.

Kowalski [5] discusses the interaction of a belief system with its environment. He proposes four possible deductive relationships between new information and an existing belief system, each suggesting different possibilities for updating the belief system:

(4) The new information can already be derived from the current belief system.

(5) Part of the information in the current system can be derived from the new information together with the information in the rest of the current system.

(6) The new information is consistent with the existing belief system but is independent of it.

(7) The new information is inconsistent with the existing belief system.

The problem is to design a system that can handle all four cases with efficiency. As the information system grows, the time and space required to determine which one of the four deductive relationships apply becomes larger and larger. It may eventually become necessary to assume the new information to be independent (6), simply due to resource limitations preventing a more definite answer.

There is a need for non-monotonic reasoning since many interesting problems cannot be dealt with using purely monotonic reasoning. Reinfrank [7] has discussed this by analyzing the situations in which problems exhibit one or more of the following characteristics:

(8) The information is unreliable and possibly inconsistent.

(9) The information is incomplete.

(10) The problem domain is instable, such as with time-varying environments.

Doyle [1], and a number of other researchers such as Goodwin [4], have proposed implementation approaches that support most of the actions required to handle the above mentioned problems. Doyle's *Truth Maintenance System* (TMS) is a problem solver subsystem for performing these functions by recording and maintaining the reasons for program beliefs. It appears that the TMS approach is an excellent foundation for a system that will be responsive to the points previously made.

Michalski and Winston [6] have used the expression *variable precision logic* (VPL) to represent methods to deal with problems of reasoning with incomplete information and resource constraints. They propose mechanisms for handling trade-offs between the precision of inferences and the computational efficiency of deriving the inferences. It is the intention that questions such as the following may be answered (as quoted from their paper):

(1) Give me a reasonable answer immediately, even if somewhat general; if there is enough time, give me a more specific answer.

(2) Give me a reasonable answer immediately; if there is enough time, tell me you are more confident in the answer or change your mind and give me another, better answer.

Non-monotonic logic investigates the formal implications of the problems with revising beliefs and modifying tentative knowledge. It ignores the time and memory resources required to support the reasoning steps used. However, systems such as a TMS are concerned with the representational and algorithmic problems of maintaining a consistent set of beliefs in the face of changing premises. In addition, VPL type systems are also concerned with the trade-offs between the certainty and specificity of decisions and the computational resources needed to derive them. The work discussed in this paper reflects the TMS/VPL paradigm.

3. The Approach

3.1. Representing the World

In most recent truth maintenance systems, the Closed World Assumption (CWA) [8] is used to reason about data that is not represented. The CWA incorporates the assumption that its information is complete. Thus any information which is not represented and which has no proof is given the negation of its desired truth value.

Within the Open World Assumption (OWA), all existing information is represented as true or false [8]. If the information does not exist and no proof is found, then no explicit assumptions are made as in the CWA. Instead the information remains unknown.

3.1.1. Extended Open World Assumption

The Extended Open World Assumption (EOWA), allows a broader representation of the world than both the CWA and the OWA. As in the OWA, facts will be represented in the EOWA as explicitly true or false. Queries to the database will return true or false if the data is represented, or unknown if the data is not represented. The extension to the OWA is that reasoning about unknown data will be incorporated in the form of assumptions. Also the EOWA will have the ability to categorize its information (Figure 1), thus giving the information

added distinction beyond the value of true, false, and unknown.

Querying within the EOWA begins a series of deductions to establish the value of the belief as represented within the world. If a belief cannot be deduced, then its negation is checked for proof. If its negation can be deduced, then the negation is asserted into the database. If its negation cannot be deduced, the belief remains unknown, due to insufficient information.

At any point in time a snapshot of the world representation within a belief management system using the EOWA can be divided into the following sets of information (Figure 1):

a. The rules that comprise the knowledge base for the world.

b. The premise beliefs (or facts) that are explicitly known to exist in the database. These premise beliefs have been entered into the database as input.

c. The beliefs that can be computed by applying rules in the knowledge base to the database.

d. Assumptions made when reasonable through the application of knowledge.

e. Everything that is not represented in one of the above parts, i.e. unknown information.

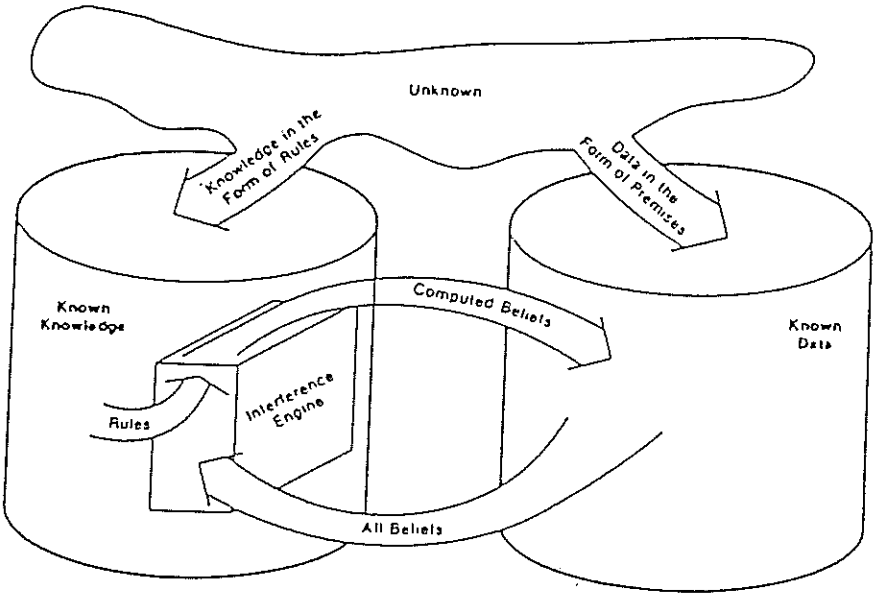

Figure 1: Working Within the EOWA

The addition and deletion of facts, querying the database, and the passing of time can change portions of the represented world.

3.1.2. Levels of Belief

Each type of belief represented in the EOWA is a member of a specific set called a level of belief. There are four levels which form a conceptual hierarchy of support [2]. The higher the level number, the more direct support the belief has. Each level is given a number from 0 through 3 (with 3 being the highest) which determines its position in the hierarchy. The levels are defined by their dependencies on each other. A belief can be a member of only one level at a time, hence making the levels of belief disjoint. In each level of belief the value of the belief can be either true or false but not both. The level in which each member is not dependent upon any other belief is called *premise beliefs* and every member is called a premise. These beliefs must be explicitly stated as true or false by the user or other external means. This level is ranked at 3, therefore it is considered the highest level.

Computed beliefs are those beliefs which have been computed using the inference engine with rules from the knowledge base and other beliefs from the database. This label refers to two disjoint levels of belief, *derived beliefs* and *inferred beliefs*. The level of derived beliefs is the set in which each member has been computed without using assumptions, making their only dependency on premise beliefs and other derived beliefs. This level is ranked at 2. At the level of inferred beliefs, level 1, each member has been computed using one or more assumptions, making inferred beliefs depend on all beliefs and at least one assumption. All inferred beliefs must be computed from a *well-founded support*. The definition of a well-founded support in this context is a premise, or any belief that has at least one premise on which its computation rests. Thus, inferred beliefs cannot rely only on assumptions.

The level of *assumptions* is at the lowest level in hierarchy, level 0. This level consists of beliefs that have been assumed at some point in time for assertion of a inferred belief into the database. In the current system, all assumptions must be authorized by the user of the system. In the future, other way to determine assumptions will be utilized. A particular assumption is dependent upon the existence of the other beliefs needed for a proof in conjunction with that assumption. If a belief is not a member of any of the above levels of belief, then it is considered an unknown belief.

As stated earlier, a belief is asserted into the premise level if it is entered explicitly by the user or by any external means. Beliefs are entered into the derived level when they are consequents of forward chaining. Inferred beliefs are entered into their level through backward chaining, when the belief is not already in the premise or derived level. An assumption is also entered into its level when needed and authorized. The uses of forward and backward chaining will appear later in sections 3.2.2.2 and 3.2.2.1 respectively.

3.1.3. Rules and Restrictions Governing Level of Belief Membership

Every belief will be a member of the highest level of belief possible in the hierarchy. It will rise to a higher level of belief at the first possible moment and if expelled from that level of belief, the system will check for possible membership in a lower level of belief before making the belief unknown by deletion. Each level imposes different restrictions as to when a belief can change its truth value. For a belief at the premise level, the restrictions are the most severe. If a premise could have the same truth value at the derived level, i.e. it has also been computed, then the world representation would be considered corrupt if that premise changed its truth

value. Also, the same type of corruption would occur if the changing premise were to contradict another premise which has also been computed at the derived level, or even contradict a derived belief, since the beliefs at the derived level are deduced using the modus ponen inference rule. The deletion of a premise belief can cause corruption of the world representation, if it is deleted under the same circumstances that cause corruption during a truth value change. This corruption implies that the rules governing the known world are not consistent with the present data and/or the other rules in the knowledge base. User intervention is sought if such a situation occurs.

The value of a member of derived beliefs can only change when a premise on which it depended changes or is deleted, and that derived belief has only one path of computation. The world representation will be considered corrupt if the derived belief whose value changed is able to be computed to its original value at the derived level in another manner. In other words, corruption would occur if an attempt was made to assert a derived belief that already existed with the opposite truth value.

Moving down the hierarchy, the restrictions on value changes lessen. The biggest difference between the ability of a derived belief and an inferred belief to change values is the use of assumptions in the computation of the value of the inferred belief. A contradiction of values between an inferred belief and any other premise or computed belief will force a change in previously made assumptions until the contradiction is resolved. This means that an entering premise or computed belief can contradict an inferred belief with corruption of the world representation, only if the contradiction cannot be resolved. If no change in the values of the necessary assumptions can be made, then corruption would exist. Thus, the only restriction made on the computation and value change of an inferred belief is that it cannot attain membership at the inferred level because of a previously made assumption about itself.

Assumptions can change values easily. Since they are used to support the truth of a belief, they can be changed easily if a contradiction arises. As long as an assumption can be the culprit of a contradiction, no world representation corruption is said to exist. Also an assumption may become unknown if it is no longer needed for the proof of a belief, i.e. if the belief becomes a premise or derived belief. It may become unknown or changed to the opposite truth value after the inference engine takes a "closer" look and decides against the original assumption that was made. If a contradictory value is entered into the data base the consequences of changing an assumption are removing its dependents and reconfiguring the levels of truth. Whenever a belief is asserted into a level of truth with whatever value, it is considered to change from unknown. Since assumptions are not computed, but authorized by the user, they are not considered to have a well-founded support. Being assumptions, they intuitively are not well-founded, thus they can only be used with beliefs that have well-founded supports in the computation of an inferred belief.

3.2. Dynamic Dependency Net

3.2.1. Definition and Use

A dependency network is a well known tool for representing and reasoning about a level of truth of beliefs within a belief management system. In the past, dependency nets were created a priori for all possible beliefs which a system could hold. Marks were then placed on the net to indicate which beliefs were currently "in" or known and which were "out" or unknown. Reasoning about beliefs consisted of propagating these marks through-out the net [1].

A dynamic dependency net is used for a similar purpose, but has a somewhat different representation. A node in a dynamic dependency net is considered a support for a computed belief. It represents the instantiation of a satisfied rule. There are no markings of "in" or "out". This support, created when a rule is satisfied, consists of three parts; the antecedent, the consequent, and the justification. The consequent is the belief that will be asserted into the database provided a node in the dependency net can be built by the creation of a justification. In order for this to occur the node structure must have a well-founded support, which is determined by the antecedent. The antecedent contains a set of beliefs and the values needed by each belief. If a premise is not a member of this antecedent, then at least one belief in the antecedent must have a well-founded support. If the beliefs in the antecedent exist in the database and have the correct truth values then a justification is created to link the antecedent to the consequent to represent a valid rule and the consequent is asserted into the database.

The network structure is used to represent the inferencing inside the world and is responsible for the representation of associations and dependencies between premises, computed beliefs and assumptions for proof of those beliefs. Because it is considered to be dynamic, it can change its representation as the world changes through queries, premise additions, and premise deletions. Though its representation reflects all those beliefs that are at the premise and derived level, it stores only those beliefs at the inferred and assumed levels that have been queried and proven by the use of assumptions. In essence, only the minimal assumptions and inferred beliefs are stored in the dependency net. The reasoning behind this action is to minimize contradictions and the handling of those contradictions found, since most contradictions will be due to wrong assumptions. The dynamic dependency net structure can give reasons for the existence of all beliefs represented.

Thus, a dynamic dependency net is allowed to grow and shrink as necessary in order to contain only those beliefs which are currently known at any point in time. As new information is discovered the net will grow to incorporate it. As information becomes invalid for any reason, it is removed from the net.

3.2.2. Net Modifications

A simple dependency net containing beliefs in each level of truth, along with justifications for the computed beliefs, is shown in Figure 2. Each level of truth is represented by a specific design. Justifications connect antecedents to consequents using an AND gate symbol. Notice that once an assumption is used within a support network, no computed consequents using that support can be members of the derived level of truth of beliefs unless later asserted as such through forward chaining, at which time the assumption is no longer needed. Querying for a belief, adding premises, and deleting premises will modify the net's representation of the world. The following sections expand on what we expect to occur theoretically during any of the modifications.

3.2.2.1. Result of a Query

The user queries the database using a basic query "is this information true?". The belief in question is checked for membership in one of the levels of truth. If the belief is present (a member of any level with a desired value or a contradiction (a member of any level with the opposite value of that which is desired), its level and value will be returned to the user. The user can then respond in two ways: (1) it can accept the information as it stands, or (2) it can force the belief to be reproven in hopes of strengthening existing supports or in the case of an assumption, possibly create supports.

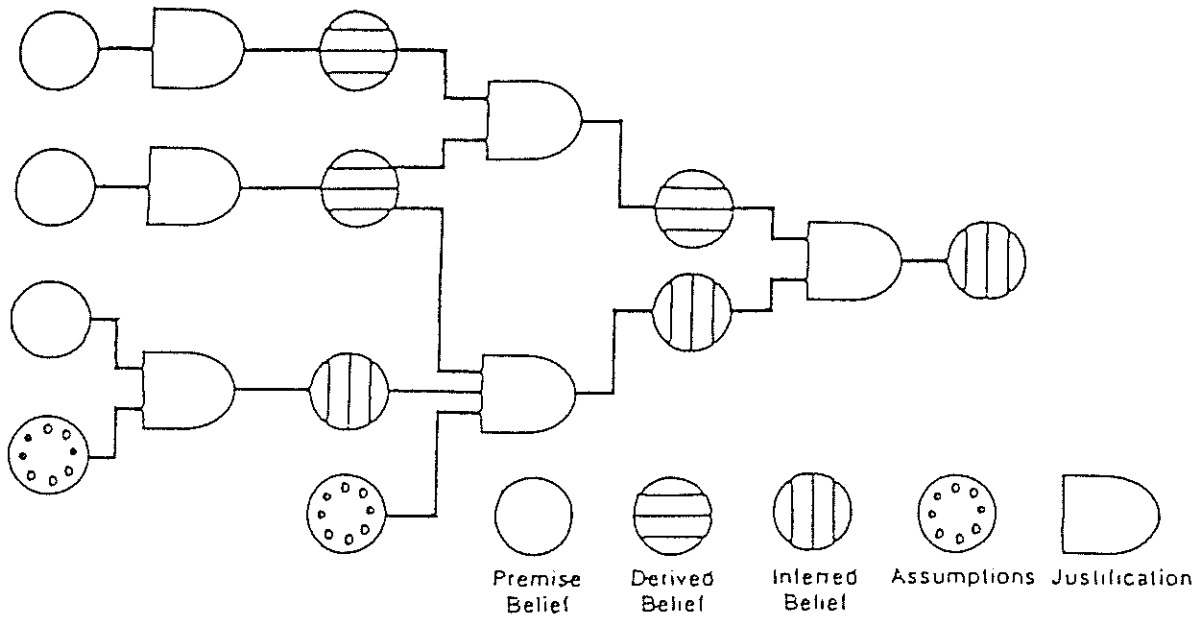

Figure 2: Representation of the Net Structure

If the belief is nonexistent (not a member of any level), an attempt will be made to prove the belief. If corresponding knowledge exists to support the belief then the deduction mechanism will backward chain through the supports and their antecedents until it can find existing beliefs in the database and begin connecting a network of supports to the belief in question. If connection can be made using existing beliefs and necessary assumptions back to the belief in question then the final value of the belief at the inferred level will be reported. The proven belief may have either the desired truth value or the opposite truth value. If no connection can be made with the resources available then "cannot be proven" will be reported. The difference between the response "insufficient information" and "cannot be proven" is that "cannot be proven" means that an exhausted proof was not able to be performed. "Insufficient information" is reported only when there is no corresponding knowledge to complete the proof.

It is possible that a change in value of an existing assumption may be necessary in order to complete the proof for the given belief. The user is then given the option to change this value, possibly destroying other inferred dependents.

Figure 3 represents what happens when a query is made. In this case the query can be proven and the necessary supports are added to the net. The belief being queried is represented by the surrounding triangle. Searching through the knowledge base, the system finds the belief as a consequent of a rule and backward chains to see what antecedents are needed to satisfy the rule. It finds it needs beliefs A and B to be present. Belief A is present at the derived level, but belief B is unknown at the time of the query, so again the system must look for corresponding knowledge in the knowledge base. This is the essence of backward chaining. Belief B is a consequent of a rule whose one antecedent is present at the premise level, but whose other antecedent is unknown. Since there exists a well-founded support, the nonexistent antecedent can possibly be assumed. Authorization is granted to the system to make the

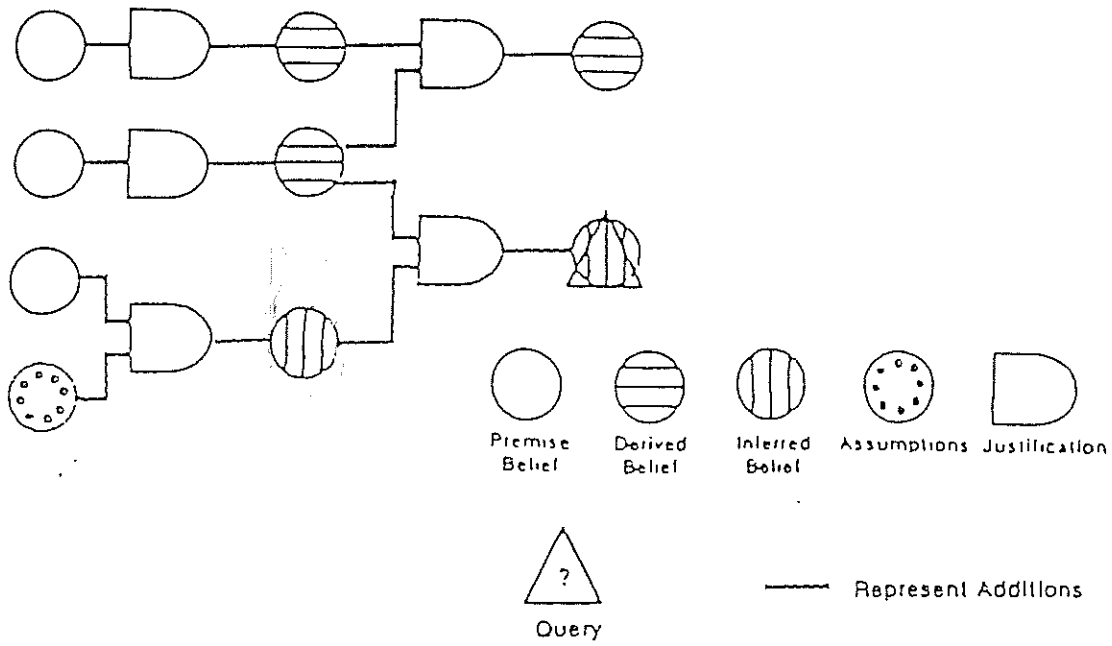

Figure 3: Result of a query

assumption and a justification for the rule can be created since the rule is now satisfied. B is then asserted into the level of inferred beliefs in the database. Once this occurs the initial rule is satisfied. Hence, a justification is created and the queried belief is added to the level of inferred beliefs in the database.

3.2.2.2. Result of Premise Addition

Any fact explicitly entering the system by external means is considered a premise and becomes immediately a member of the level of premise beliefs. Entering premise values have priority over all existing values of the same belief at any level, unless corruption occurs as discussed in section 2.1.3. The same belief will be expelled from any level lower on the hierarchy in order for the premise to be asserted anew. The first step in premise addition is to check every level for membership of the entering belief.

If the belief is present (it is a member of any level and its value is the same as the value to be asserted) then it is expelled from any level other than the premise beliefs and asserted into the level of premise beliefs. If it is a contradiction (it is a member of a level but has the opposite truth value), then corruption is determined. If there is no corruption, then any contradiction is resolved, and the forward chaining process is performed to build added support using existing knowledge and beliefs.

The forward chaining procedure attempts to satisfy as many rules as possible using the modus ponen inference rule. It is limited to the use of only those beliefs at the premise and derived level. This includes the entering premise and all asserted derived beliefs. There are no inferred beliefs or assumptions involved in the forward chaining process.

If the entering belief is nonexistent (it is not a member of any level), it can be asserted immediately and forward chaining begins. If at any time the forward chaining causes a contradiction to occur in one of its computed beliefs then corruption is determined. If no corruption exists then the contradiction is resolved. It is possible that the forward chaining process causes beliefs to be asserted at a higher level. For example, a derived belief may be able to take the place of an inferred belief of either the same or opposing truth values. If this occurs then the net must be relabeled. This short procedure will be discussed in section 3.2.3.

Forward chaining within a dynamic dependency net can cause destruction of supports when a contradiction is reached and a justification becomes invalid. In Figure 4, belief A is an inferred belief having a value of true in the existing net structure. The premise is added, surrounded by a rectangle, and the forward chaining attempts to assert belief A into the level of derived beliefs with a false value. Because of the conceptual level hierarchy, the belief should be asserted the derived level. But the contradiction of values needs to be resolved first. Thus, belief B, an assumption is retracted causing destruction of supports that depended on the value of that assumption, and A is free to be asserted at the derived level.

3.2.2.3. Result of Premise Deletion

In the EOWA, some premise beliefs may no longer be known to exist even though they have been explicitly asserted. Thus, the system allows for the deletion of such premises. One action that may be taken after a premise is deleted is that the lower levels of truth are checked for possible membership of the deleted premise. If found then the value found is asserted for that belief into that level. This is done by the relabeling procedure in section 3.2.3. If the belief

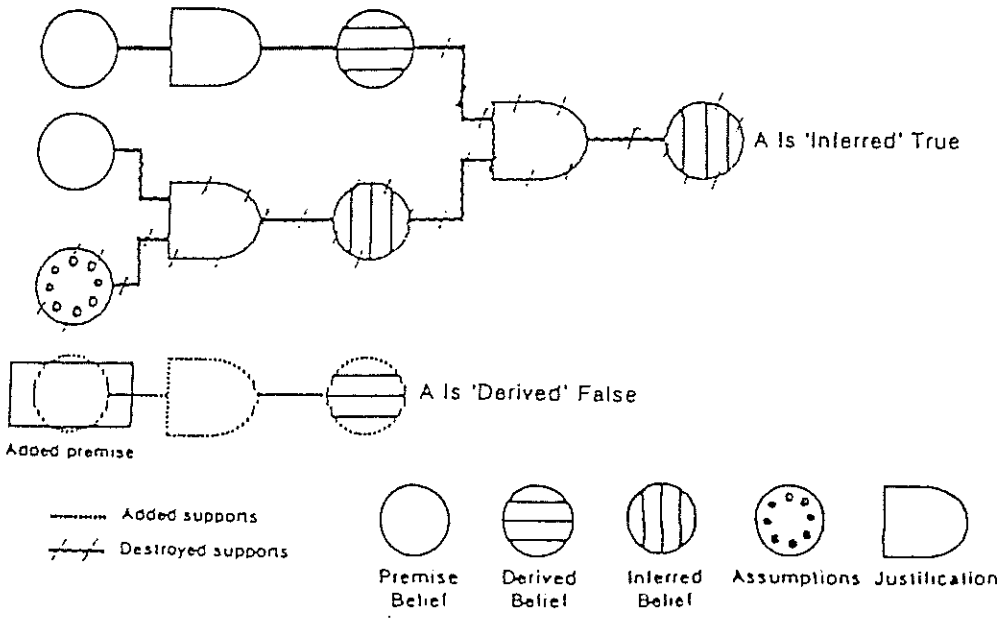

Figure 4: Result of premise addition

is not found to exist in any other level, then all justifications using this premise will become invalid. The consequents of those justifications may be removed unless they have alternate support.

A simple representation is in Figure 5 in which a premise is deleted from an existing network of beliefs, as depicted by the smaller circle inside the premise representation. This occurrence causes invalidation of those supports dependent on the existence of that premise. This is in turn causes retraction of beliefs that were consequents of the destroyed supports.

3.2.3. Relabeling

Relabeling refers to the membership switching of beliefs among levels. The relabeling procedure is needed when a belief rises to a higher level. Its consequents, if any, may also rise to a higher level. This rising of beliefs may cause unnecessary assumptions to be present in the database. The relabeling procedure will remove these assumptions as they become obsolete. Relabeling is also performed when a premise is asserted in the place of a belief at any lower level than itself, and also when a derived belief is asserted at any lower level than itself. Thus, the procedure can work the opposite way. If a premise is deleted and can also reside at the derived level, then that premise and its consequents may be relabeled. The same would occur if a premise or derived is deleted but may reside at the inferred level.

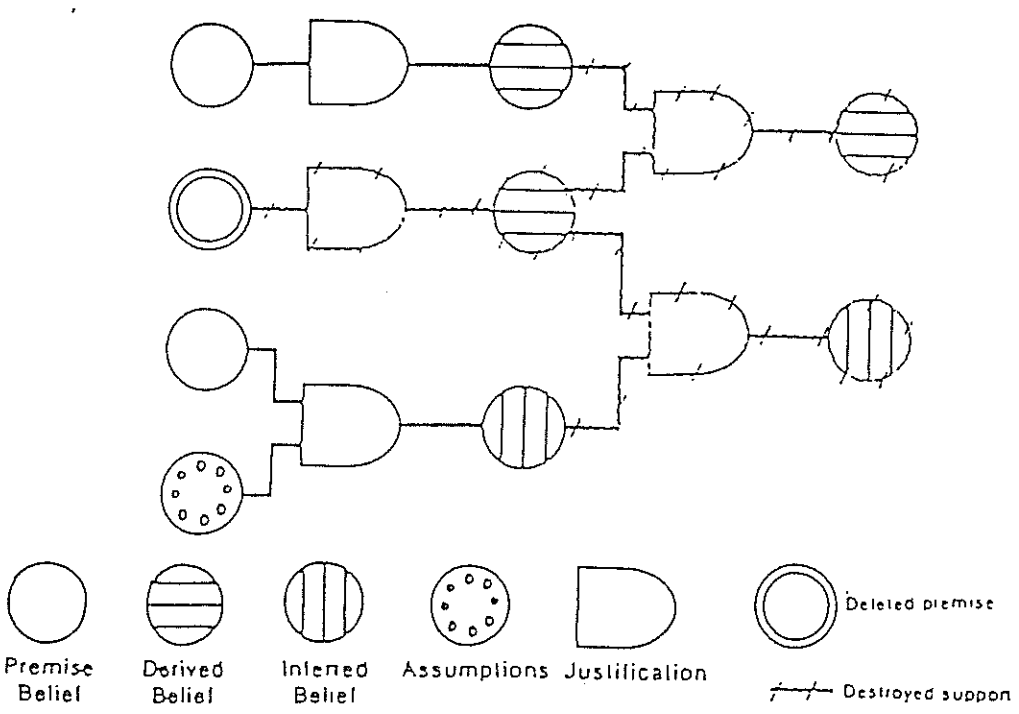

Figure 5: Results of premise deletion.

4. Discussion

The following section is a brief discussion on how our first implementation handles the problems presented in the background section of the paper.

Three basic problems in dealing with deductive databases were presented in section 2, points (1), (2), and (3). In this first BMS, negative information must be explicitly entered in the same manner that positive information is entered, i.e. we can enter "X is true" and "Y is false". During a query, the system attempts to deduce a belief with a given truth value, using rules from the knowledge base and other beliefs from the database. If the system cannot deduce the belief to have the desired truth value, it attempts to deduce the belief with the opposite truth value. Because the system needs to do this type of deduction, an attempt should be made to engineer a rule, whose consequent has one truth value, to produce a sister rule whose consequent has the opposite truth value [2]. The null value problem is dealt with by reporting that the data item is missing or unknown. Then a proof can be attempted to infer a value for the missing data item. If no proof can be established the belief remains unknown or missing from the database. The indefinite data problem is not handled at this time.

The relationship labeled (5) is handled according to the suggestions by Kowalski [5]. We have shown that if (7) occurs, unless the necessary assumptions can be altered, corruption is reported to the user. Relationship (6) deals with the new information being independent of the old information. In the current BMS, we have found no problems as of now in this relationship, and thus add the information to the database as independent information. We recognize the importance and ramifications of this relationship, making it a consideration for further research. If the new information already can be deduced, point (4), then the system remains the same. The relabeling procedure, in section 3.2.3, provides a mechanism for the removal of any unnecessary information.

It is obvious that non-monotonic reasoning is needed and used extensively in this BMS. Handling points (8), (9), and (10) are directly incorporated into the algorithms that run the system. Because of the capabilities desired, our BMS contains the functionality of a general truth maintenance system, and more. Dependency directed backtracking is used to determine wrong assumptions. In the future, a more sophisticated algorithm will be implemented, but in this BMS, all changes in assumptions must be authorized by the user. Also the concept of every belief having a well-founded support is incorporated. Each premise is considered its own well-founded support and every belief deduced or inferred must rest on at least one premise in its computation. Also a belief cannot rest on the existence of itself, i.e. a belief cannot be used in its own computation. This will inhibit any circular computations. An assumption has no well-founded support. Since it must be authorized by the user in this BMS, assumptions cannot be computed using other beliefs. Hence they rely only on the external source that created them.

5. Extensions

Since our ultimate purpose is to operate the belief maintenance system under limited resource constraints, many extensions will be made to this initial prototype. The mechanisms required to support Variable Precision Logic have been a consideration throughout the development of this system, but have not yet been implemented.

The interaction between belief maintenance and limited resources requires further investigation before any future implementation is attempted. The ramifications of limiting the inferencing done in both forward and backward chaining are great. The system will have to

maintain integrity on a larger scale to the maximum extent possible. Consistency maintenance must begin with the knowledge base during the interactive addition and deletion of rules. It will then be extended to maintaining database integrity when limited resources prevent the completion of the forward chaining mechanism, and continue with the use of Variable Precision Reasoning during querying and backward chaining.

6. Conclusion

We have described an approach to building a cognizant database that presents a significant ability to handle and process dynamic information flows. Prolog was used for the initial implementation. This language allowed easy modification, extension, and experimentation on the basic algorithms as the system evolved.

7. References

1. J. Doyle, A Truth Maintenance System,*Artificial Intelligence 12*,3 (November 1979), 231-272.

2. R. M. Fulcomer, W. E. Ball, and J. P. Tadlock, Belief Maintenance Systems, WUCS-CV-87-1, July 1987.

3. H. Gallaire, J. Minker and J. Nicolas, Logic and Databases: A Deductive Approach, *Computing Surveys 16*,2 (June 1984), 153-185.

4. J. W. Goodwin, WATSON: A Dependency Directed Inference System, in *Proceedings of Non-Monotonic Reasoning Workshop* , American Association for Artificial Intelligence, October 17-19, 1984, 103-114.

5. R. Kowalski, *Logic for Problem Solving*, Elsevier North Holland, Inc, 1979.

6. R. S. Michalski and P. H. Winston, Variable Precision Logic, *Artificial Intelligence 29*,2 (August 1986), 121-146.

7. M. Reinfrank, An Introduction to Non-Monotonic Reasoning, MEMO-SEKI 85-02, June 1985.

8. R. Reiter, On Closed World Data Bases, in *Logic and Data Bases*, H. Gallaire and J. Minker (editor), Plenum Press, New York, 1978, 55-76.