

REASONING IN INCOMPLETE DOMAINS

Steven Rosenberg
Information Methodology Research Project
Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720

Most real world domains differ from the micro-worlds traditionally used in A.I. in that they have an incomplete factual database which changes over time. A traditional rule interpreter such as Planner can be extended to construct plausible inferences in these domains by (A) allowing assumptions to be made in applying rules, resulting in simplifications of rules which can be used in an incomplete database; (B) monitoring the antecedents and consequents of a rule so that inferences can be maintained over a changing database.

Many real world databases are:

- a. Incomplete, because not all the information needed for an inference is available at a particular time.
- b. Unstable, since the particular subset of information available can change fairly rapidly in the real world.

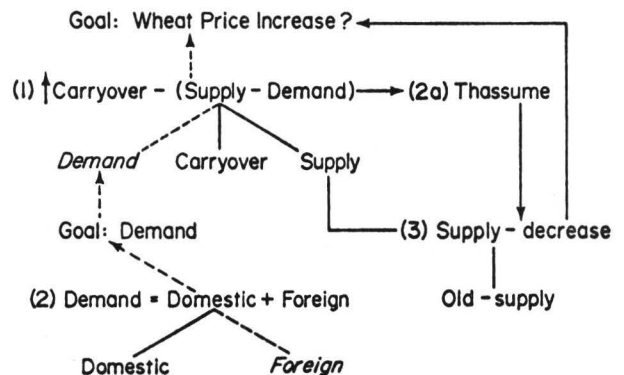
For example, a farmer, in deciding how large a wheat crop to plant, has incomplete information about ultimate wheat price and demand, based on rapidly changing information, such as daily weather reports, weekly crop surveys, and so on.

1. INCOMPLETE DOMAINS

A traditional reasoning program, such as Planner [6], has difficulty dealing with incomplete information. Consider how such a program might make inferences about the price of wheat. We, formulate the problem as follows:

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the Laboratory's artificial intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract No. 14-75-C-0643. The author is now with the Information Methodology Research Project, Lawrence Berkeley Laboratory, Berkeley, CA 94720.

(Thgoal (price-increase wheat))



(In the diagrams, italics and a dashed line indicate antecedents which are missing; a solid line indicates an existing antecedent; a solid line and arrow indicate deduced links between rules and assertions; a dashed line and arrow indicate inferences which have failed.)

Thm1 states that to show a price increase for wheat, show that the difference between supply and demand is less this year than last. Suppose that no assertions for wheat demand exist in the database. In this case, Thm1 would fail, since one of its antecedents is missing. For such cases, Planner provides a strategy. If an assertion is not in the database, Planner will try and prove it. Thm2 states that to deduce demand for a commodity, find the foreign and domestic demand for this commodity, and add these together. We will assume no assertion for

foreign demand exists. Thus Thm2 will fail. If there are no other methods for proving the missing assertion, the inference process will fail.

The inference attempt failed not necessarily because it was wrong, but because not enough information exists to make correct inferences. By making assumptions about missing information in such cases we can often succeed by using a simpler rule. For example, a farmer might reason that as long as the supply of wheat is decreasing, it will be worthwhile to plant more regardless of demand. Thus, if the full set of assertions concerning supply, demand and carry-over are unavailable, we might try to prove that supply has decreased. By substituting goals that require only a subset of the assertions that the original theorem required, we develop a notion of rule simplification.

The knowledge of which other rules can be used as simplifications, and under what circumstances, must be represented. A rule interpreter can make use of this metaknowledge [1] to substitute simpler theorems for a rule which fails. The appropriate place to specify this metaknowledge is in a separate class of theorems. e.g.

```
(Thassume Thm2A (X) (Supply-&-demand ?X)
  (Thgoal (supply-decrease ?X))
  (Thcaveat (Default)))
```

This new class of theorems, such as Thm2A, contains information concerning simplifications and assumptions. A Thassumption will specify A) a goal; theorems satisfying this goal can function as a simplification; B) the assumptions involved in using that simplification. These are expressed as a caveat. If the assumption being made is that the missing antecedents can be ignored, the caveat will contain a Default.

Using a simplification always reduces the plausibility of the reasoning. By ignoring the demand for wheat, a farmer is willing to assume that if demand changes, it will not change in a direction or quantity which would invalidate his reasoning. By making explicit this notion of assumptions, we can extend the list of options available. For example, we can encode constraints on our use of assumptions through the caveats.

Returning to our example, after Thm2 fails, simplifications will be considered, and Thm2A (proving a decrease in wheat supply) found. Thm2A first tries to satisfy the goal (Thgoal (Supply-decrease ?X)). Thm3 can be used to prove this. Thm3 states that to show a

decrease in supply, show that last year's supply is greater than current supply. If current supply and old-supply are known, Thm3 will succeed, and support the hypothesis of higher wheat prices.

2. UNSTABLE DOMAINS

As the database changes, our goals may remain relatively stable. Support for an hypothesis is conditional on the assertions available at the time it was first considered (i.e., the inferences which were possible at that time). However in many deductive systems, once deductions are made from a set of assertions, no effort is made to insure that while the results of those deductions are used, the assertions still hold true. Generally, the user does not expect the database to change so as to invalidate prior inferences.

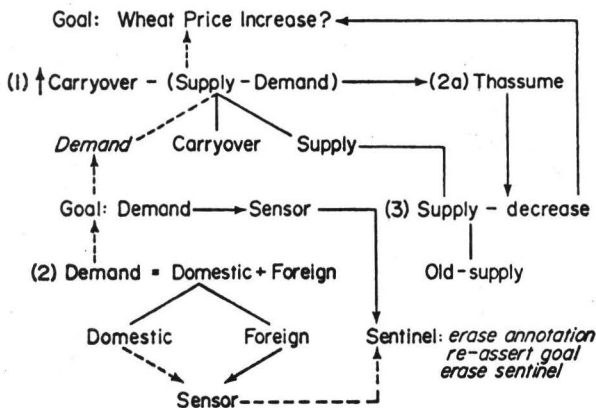
We can extend our concept of rule interpretation by making the maintenance of rule instances a function of their interpretation. We implement this by giving each active rule the autonomy to respond to changes in its environment. As each rule is applied, an associated Sentinel is created. The Sentinel knows how to respond to changes in that rule's antecedents or consequents. The result is the maintenance of hypotheses through a method of local autonomy. A rule instance must be continuously enabled while it is used in support of some hypothesis.

A Sentinel associated with a rule instance will place sensors in the database. A sensor is a demon which responds to changes in the pattern that triggers it, and evaluates whether these changes violate some criterion. If so, the sensor reports to the Sentinel. The Sentinel is satisfied when some arbitrary logical conjunction of its sensors succeed. The Sentinel can then take a variety of actions. The standard ones are to A) erase itself; B) erase the rule instance; C) reinvoke the goal.

Individual rules will succeed or fail as a function of their antecedents. When a successful rule's antecedents change, its sentinel will be triggered. When this happens the sentinel causes the goal the rule was supporting to be re-evaluated. At this point the sentinel can erase itself. (Note: A sentinel causes a goal to be re-evaluated. There is no constraint that the same rule be used again. Another rule may now be the best choice. However, in this example it is assumed that there have been no other changes in the state of the system that would cause another rule to be selected first.)

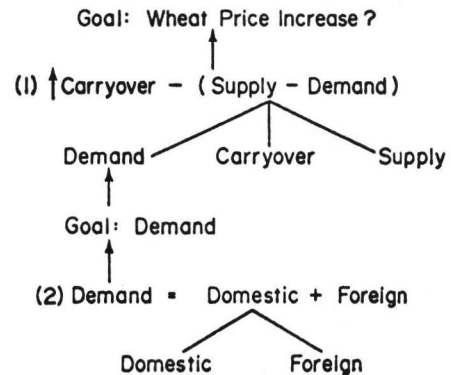
If a rule's goal is removed, its sentinel will also be triggered. In this case we do not wish to re-evaluate the goal. The sentinel will remove itself and erase the associated rule instance.

Consider how this local association of rule instances with sentinels can give rise to the right global behavior. Suppose that a failed rule now is capable of succeeding, through its missing antecedent being asserted. For instance, the missing foreign demand for wheat can be asserted.



This will trigger the associated sentinel to erase the annotation for this rule instance, reassert the goal as something to be proved, and then to erase itself. This time Thm2 succeeds, resulting in a proof of the missing demand for wheat. This will in turn trigger the sentinel associated with the rule instance of Thm1 for which the missing demand is an antecedent.

This sentinel repeats the actions of the prior sentinel. However, in erasing the annotation, it erases the record of the assumption made. This will trigger the sentinel on Thm3, the simplification. Since the use of a simplification is conditional on another rule failing, the sentinels associated with theorems used as simplifications monitor the annotation recording that failure, so that they will know when the simplification is no longer required. They will then respond to the erasure of this annotation by erasing the annotation for the simplification. Since the formerly missing antecedent for demand has now been inferred, Thm2 succeeds, as does our inference attempt.



Thus, unneeded rule instances will know when to remove themselves. Through local propagation, the representation responds to changes in the available database. Changes in the database can reinvokethe goal of inferencing, which can then proceed. Doyle [2] has developed a scheme for recording deductive dependencies so that when facts turn out to be incorrect, the entire "context" of dependent facts can be removed from the database. The current approach does not allow explicit manipulation of deductive dependencies, since the changes it is designed to respond to are those imposed on the domain by "outside" changes in the data, and not ones due to deduced inconsistencies. McDonald [3] has contrasted this approach to other methods.

The rule interpretation features of sentinels and simplifications were programmed in FRL (Frame Representation Language) [4].

REFERENCES

- [1] Davis, R. and Buchanan, B.G., "Meta-Level Knowledge: Overview and Applications." Proceedings of the 5th International Joint Conference on Artificial Intelligence, Cambridge, Mass., 1977.
- [2] Doyle, J., "Truth Maintenance Systems for Problem Solving." MIT Artificial Intelligence Laboratory Technical Report 419, 1977.
- [3] McDonald, D., "Story Understanding: the Beginning of a Consensus." MIT-AI Working Paper 168, 1978.
- [4] Roberts, B.R. and Goldstein, I.P., "The FRL Manual." MIT-AI Memo 409, 1977.
- [5] Sussman, G.J., Winograd, T. and Charniak, E., "Micro-Planner Reference Manual." MIT-AI Memo 203A, 1971.