Integrating Rules and Inheritance Networks in a Knowledge-Based Financial Marketing Consultation System

Tim Daly, John Kastner, Eric Mays

IBM Research Division
Thomas J. Watson Research Center
Yorktown Heights, New York 10598

Abstract

This paper describes the integrated use of rule-based inference and an object-centered knowledge representation (inheritance network) in a financial marketing consultation system. The rules provide a highly flexible pattern match capability and inference cycle for control. The inheritance network provides a convenient way to represent the conceptual structure of the domain. By merging the two techniques, our financial computation can be shared at the most general level, and rule inference is carried out at any appropriate level of generalization. Since domain knowledge is represented independently from control knowledge, knowledge about a particular problem solving technique is decoupled from the conditions for its invocation. A large financial marketing system has been built and examples are given to show our combined use of rules and inheritance networks.

Introduction

This paper describes some aspects of a knowledge representation and pattern match facility in a financial marketing consultation system. This facility may be characterized as a melding of an object-centered knowledge representation in the form of a structured inheritance network along with a flexible pattern match capability and rule inference cycle for control. By merging the two techniques, methods for computation are shared at the most general level, and rule inference is carried out at any appropriate level of generalization.

Our financial marketing consultation system, FAME, is designed to provide a marketing representative with an integrated set of tools to analyze a company's financial status and data processing needs, recommend alternative long-term data processing solutions for those needs, and provide advice on the various financial acquisition possibilities. We have found that the domain of financial marketing is a good vehicle for testing our ideas on knowledge representation.

The term *financial marketing* characterizes the financial decision processes used in the marketing of products and services of such large scale that they can significantly impact a company's financial status. Financial considerations often become as important as computing considerations in making very large equipment proposals. FAME makes use of multiple representation schemes and problem solving algorithms.

Representation Facilities

Our basic representation and reasoning facility, KROPS, may be characterized as a combination of an object-centered knowledge representation, K-Rep,[1] with a flexible pattern match and rule system, YES/OPS[2]. This gives KROPS a dual character. Viewed from the YES/OPS side, it is a pattern matching language on an inheritance network. Viewed from the K-Rep side, it is a knowledge representation system with pattern sensitive procedures. These two facilities are made available to the system developer as either separate AI programming tools or as a highly integrated system.

K-Rep is based on the tradition of frames and semantic nets (e.g. KRL[3], KL-One[4]). Unlike most frame-based systems, K-Rep incorporates an *enforced semantics* in the style of KL-One, which enables the system to do automatic classification. That is, we allow more specific objects to specialize properties of the more general, but never to override a property.

The basic unit of representation in K-Rep, as in KL-One, is called a *concept*. Concepts may be specializations of other concepts, in which case the more specific concept inherits from the more general. Specific instances of a concept are called *individual* and a concept which potentially can represent a class of instances are called *generic*. Individual concepts are always leaf nodes in the inheritance network. Attributive information about concepts is given via a binary relation, called a *role relation*, to some other concept. In addition, unrestricted information may be attached to concepts and roles using a *facet*. Facets have been introduced in a number of frame-based systems[5] as a mechanism for describing properties for slots.

A unique feature of the K-Rep classification facility is that as concepts are added, modified, or deleted, the concept network is automatically restructured to maintain the consistency of the knowledge base. This reclassification provides a powerful facility for automatically dealing with aggregate objects and multiple levels of abstraction.

YES/OPS is based on OPS5,[6] a rule based programming language. YES/OPS, like OPS5, a discrimination network called a RETE, to represent its production rules. The RETE contains partial match information for the various antecedent clauses of the rules.

The basic unit of representation in YES/OPS, as in OPS5, is called a *working memory element*, or WME. WMEs are basically lists of attribute-value pairs. As a WME is added, modified, or deleted, the RETE of partial matches is automatically updated to reflect the change. Assuming that changes are localized in the RETE, this process of constantly keeping partial matches is very effective in speeding the pattern matching process.

Features provided by YES/OPS but not OPS5 include the ability to perform pattern matching in the consequent part of the rule, the ability to freely mix in arbitrary LISP expressions with any part of a rule, and significant performance improvements.[7]

Our use of a combination of an inheritance network and a RETE represents a gradual and natural extension of our project. Originally, we used an OPS RETE exclusively[7]. As the size and complexity of our implementation increased to provide greater domain coverage, we increasingly needed capabilities to deal with aggregate objects and abstractions. Parts of our domain are highly structured and it is both computationally efficient and characteristic of human experts in the domain to deal directly with those structures.

Flexible Problem Solving Control

One of the most interesting properties of financial marketing from the viewpoint of computer science is the lack of a single answer. In fact, more important than an answer is the ability to provide alternatives along with appropriate justification. In a typical session, the user* will want to generate possibilities, to experiment with "what-if" scenarios, one leading to another. The ill-defined nature of this search space along with the many hidden variables involved in financial marketing make a specification for automatic search difficult. Instead the use of flexible control seems most promising. Therefore, we allow the user to abort any line of questioning, establish backtrack points, focus on specific items of concern, or answer "unknown" at any point.

We make extensive use of K-Rep to represent the current state of problem solving. Multiple problem solvers using different analysis methods use this representation for common communication. The explanation facilities use this representation to generate appropriate output. The control mechanism uses this representation to guide and focus the problem solving activity. The K-Rep structures built dynamically during a consultation session represent the input by the user, databases, and various problem solvers, the intermediate results obtained by the system, and the conclusions. Unlike a static trace of program activity, the K-Rep concepts are dynamic objects that heavily use inherited roles, functional attachments for inferred roles, and developer defined facets. The automatic reclassification algorithm moves the concepts around as the context changes thus providing a change in the description. The various problem solvers invoked during a consultation can examine the current state of the knowledge base represented in K-Rep to determine the current context.

Consider the following simple example:

> *A customer currently has an IBM 3081 KX3 processor installed, and is close to exceeding the capacity of that machine. He could select a larger processor from the same product family as the existing machine, such as an IBM 3084 QX6, or, alternatively, from a product family that represents newer technology, such as an IBM 3090 200. The 3081 KX3 is still on lease, so issues of lease termination penalties could be important. The leasing firm through which the 3081 KX3 was acquired may be offering a "no penalty" termination of the lease on the 3081 KX3 if it is replaced by a 3084 QX6 leased from the same firm. On the other hand, the 3090 200 will have a higher residual value, and may have other financial incentives as well. Also, it may be important to know which person will make the acquisition decision, and whether that person is measured on budget, earnings per share, or some other financial criterion.*

In a typical consultation session for this situation, the user might direct FAME to compare two ACTION-PLANs. Each ACTION-PLAN will consist of a set of ACTIONs. Each ACTION is a decision to ACQUIRE or DEACQUIRE (remove) a computer along with the supporting components such as power distribution units, processor controllers, and consoles. These are collectively referred to as the central electronic complex (CEC). Each of these entities is modeled as a *generic concept* in a K-Rep knowledge base. During the consultation, instances of generic concepts, such as ACTION-PLANs and ACTIONs, are created through dialogue with the user, reference to various databases, or activities of various problem solvers.

Figure 1 shows some of the K-Rep structures created during a typical marketing consultation. The CURRENT-PROBLEM concept points to the various major concepts representing the problem description. The figure shows that two alternative plans are being considered, each with a single acquisition ACTION. ACTION-1 specifies the acquisition of a new IBM 3090 model 200 mainframe computer while ACTION-2 specifies the acquisition of a used IBM 3084 model QX6 mainframe. Notice that these instances are subsumed by the very general ACQUIRE concept.

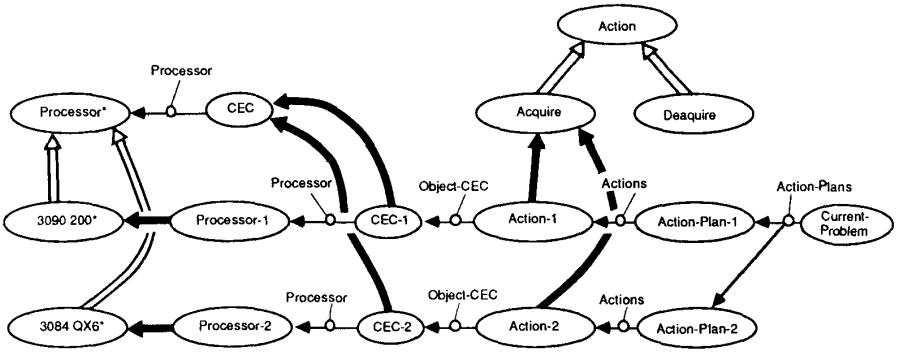

Figure 1. Portion of the K-Rep network during a consultation*

Our system has been designed for the IBM Marketing Representative.

In this and the following figure, a (generic or individual) concept is represented by an ellipse. A wide arrow (which has not been filled in) from a concept A to a concept B denotes that the generic concept B subsumes the generic concept A. A shaded wide arrow from a concept A to a concept B denotes that the generic concept B subsumes the individual concept A. A role is represented by a narrow *link* leaving a concept, and pointing to the role's value.
Some of the examples have been simplified for illustrative purposes.

The specifics of how each acquisition is to be accomplished are presently undecided. This is the state of affairs as the Financial Acquisition Problem Solver is given control.

Given the context described in Figure 1 with specified ACTION-PLANs and ACTIONs, a specialized rule-based model of acquisition/deacquisition matching and selection is invoked. The following KROPS rule* uses both the K-Rep network (current-problem) and a temporary OPS variable (goal).

```
(defrule analysis when
  (current-problem
     action-plans.local-value= <ap> & ne nil
     customer= ne nil)
  (goal <goal>
     next-step= analysis)
  then
  (modify <goal>
     next-step=
        (if (financial-analysis)
            'calculations
            'alternatives)))
```

This rule tests that the CURRENT-PROBLEM concept has been created and that its role ACTION-PLANS has a non-NIL local value (i.e. there is an instantiated action plan). In addition, the second clause of the rule tests that a GOAL to perform the financial analysis has been requested. The consequent part of the rule (after the THEN) advances the goal step depending on the outcome of the financial analysis. This rule shows the free intermingling of K-Rep concepts, YES/OPS working memory elements, and LISP expressions.

The financial planner determines how each individual action in the plan should be carried out. We employ a special heuristic classification to place these individuals in K-Rep under more specific acquisition/deacquisition methods concepts (e.g. LEASE, PURCHASE, LEASE-THEN-PURCHASE). This capability to use rule-based classification is useful for sub-problems where K-Rep's definitional classification alone would fail due to incomplete states of that portion of the knowledge base which pertains to the current problem.

In Figure 2, the ACTIONs are shown after being heuristically classified by the acquisition planner. New role information (not shown) has been added to these ACTIONs by the financial planner giving the decision criteria used for the classification choice. Under the same queries as before, these ACTION instances can now provide more specific information since they inherit from more detailed acquisition concepts. The financial planner has determined that the new 3090 200 should be purchased outright, while a used 3084 QX6 should be acquired as a conditional sale.*

We have found that multiple problem solving approaches are used in the analysis of a marketing situation. For instance, the evaluation of a customer's computer capacity growth could be based on current machine utilization, historical utilization, historical purchases, long-term business plans, peer group analysis, or various combinations of these. Each analysis requires different input, some quite time consuming to obtain, and has different customer dependent reliability. The choice of analysis method or methods is directly tied to the justification required for the proposal, the size and complexity of the proposal, and the length of the analysis period. In the field today, a wide variety of analytical methods are utilized. To provide a system for general use requires that the choice be under user control.

The problem solving control allows the user and the various problem solvers to abort any line of analysis or questioning. The present state of problem solving is available at that point represented as concepts in the K-Rep network. The set of appropriate actions is decided by the use of pattern matching on those concepts along with temporary goals previously established by the problem solving control.

Features of the Representation

The KROPS language supports a free intermingling of all of the features of K-Rep and YES/OPS. However the user is not required to use the combination. KROPS may instead be treated as a tool kit. There is extensive overhead in the maintenance of the inheritance network. Automatic reclassification of concepts in the network can represent a substantial computational burden. There is often no need to classify temporary variables although the use of pattern matching on these variables is desirable. Therefore we allow

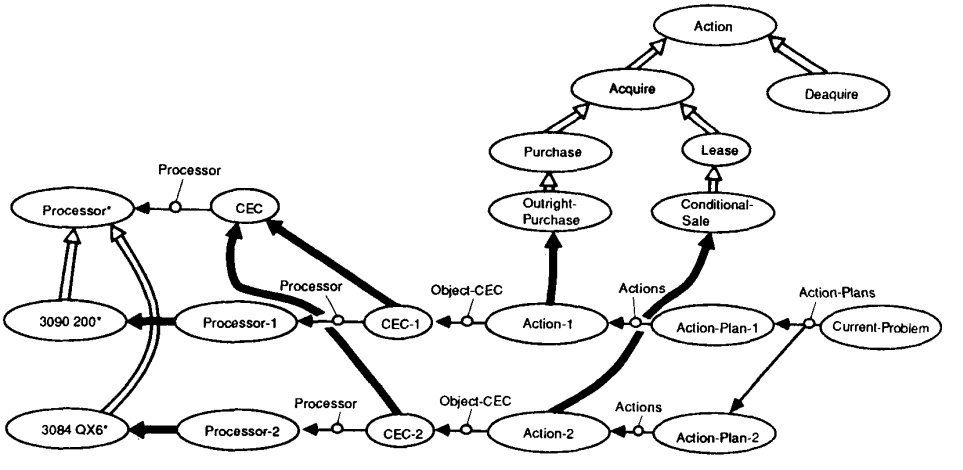

Figure 2. Portion of the K-Rep network after acquisition planning

*The financing methods chosen for the example are for illustrative purposes only.

that any variable may be declared as an OPS working memory element instead of a K-Rep concept. The pattern matching capabilities and syntax is substantially the same for each.

The integration of YES/OPS and K-Rep was performed without impacting the RETE match in any significant way. When only working memory elements are being matched there is only a type determination in the run time lookup of values. Attributes in condition element patterns are compiled into array indexes but role names are passed along unchanged and the value restriction is looked up at run time. This is done to preserve the flexibility of the inheritance network allowing late binding of values.

K-Rep Features

K-Rep adds significant representational power to the OPS class of languages. K-Rep provides convenient facilities to represent a hierarchy of concepts. We have found this very useful to represent abstractions and aggregations since the automatic reclassification of concepts insures a consistent network.

KROPS allows one to write rules that directly match abstractions and aggregations. Thus, there is potentially a great efficiency gain because we typically find that such generic concepts usually represent hundreds of individuals. In addition, the K-Rep inheritance mechanisms provide great economy in structuring our domain representation. Our previous implementation of FAME (Kastner et al. 87) incurred substantial computational complexity disadvantages by dealing directly with grounded instances. Further, we find that our human experts often deal with abstractions. We find an inheritance network representation amenable for expressing these concepts.

K-Rep allows the attachment of unrestricted expressions to concepts and their roles through the use of facets. We access facets on each CONCEPT or ROLE slot with a *dotted accessor* that will cause a binding of the destructured value. The following rule binds the variable <d> to the DOCUMENTATION facet of the CONDITIONAL-SALE concept

```
(defrule describe-conditional-sale
   when
   (conditional-sale *.documentation= <d>)
   then
   (say <d>))
```

When this rule runs, the documentation for conditional-sale is output

```
A conditional sale treats the lessee
(user of the asset) as the owner from
an IRS tax viewpoint.  That is, the
lessee is entitled to any tax benefits
(depreciation, interest, deduction, and
investment tax credit, if applicable)
even though title has not passed to the
lessee.
```

We also allow similar unrestricted attachments to roles

```
(defrule get-processor-floor-space
   when
   (3090-600e footprint= <f>
      footprint.units= <u>)
   then
   (say <f> <u>))
```

which would output

```
171.7 square feet
```

Here the role FOOTPRINT (minimum required floor space for the processor) with value 171.7 also has the facet UNITS which has the value "square feet".

YES/OPS Features

KROPS adds flexible pattern matching to the inheritance network representation by allowing matching of arbitrary patterns over concepts. The KROPS user has the ability to attach procedures to arbitrary pattern of concepts using user-defined LISP predicates. Further, the YES/OPS component allows ordinary OPS style working memory to coexist with concepts within the RETE pattern match. Thus, temporary data or control data need not be embedded within the domain knowledge.

The sensitivity to patterns within the knowledge base allows one to control the flow of interaction with the user of the system in quite natural ways. The system is able to recognize the existence of multiple ways of proceeding based on the available information and choosing the most appropriate one.

Further, this knowledge sensitive control can be freely intermixed with a state machine style of user interface. The interaction with the user can be designed as state transitions but the states can be decoupled and reordered as available information changes. Thus, the states more properly follow the AI planning paradigm of preconditions and postconditions.

Discussion

KROPS allows the attachment of arbitrary procedures in both the K-Rep knowledge base and the rules. Since K-Rep procedures are intended for use independently of the problem solving context, the results may be cached to increase response on subsequent access. Rule procedures in the rule consequent are always performed on each access, thus safely allowing arbitrary dependence on global variables and problem solving state. We have achieved substantial performance improvements via the judicial use of cached values.

Implementation Considerations

In OPS style languages, when a new working memory element is created, the newly created partial matches are propagated through the RETE. One of the first design problems that arose when connecting a RETE to an inheritance network was the choice of the structure of the data passing through the RETE. Our OPS code uses a vector representation of each working memory element and the compiler optimizes accesses into vector indices for efficiency. The indices are computed based on the working memory element declarations. Since our K-Rep network doesn't have a static declared structure, vector access cannot be used in this way. We generalize the vector index so that when a concept rather than a working memory element is given, a run-time lookup is used. However, for efficiency reasons it is important to preserve the fast vector access generated by the compiler for working memory elements.

At rule definition time it is determined whether a condition element refers to a working memory element or concept. Thus, if a RETE node contains tests for working memory, it will perform an efficient but fixed vector index. If the RETE node contains tests for a concept, it will perform a function call that is less efficient but more flexible. Whenever there is a change to working memory or to the inheritance network, the appropriate partial match changes are propagated through the RETE.

Ongoing Research

We have recently added several new concept types to K-Rep. Among these are time and set concepts. We now explicitly represent some of the time varying qualities of product pricings, financial offerings, and legal term and conditions. We use sets to represent groups of possibilities. We currently don't have efficient pattern match or iteration capabilities for these concept types.

The 'dotted accessor' mechanism originally used for facets has been found to be much more general than was first perceived. As we have experimented with new types of concepts and roles, we have found that this mechanism easily allows extensibility of the rule language.

Finally, we are considering a representation of the rules themselves within the K-Rep network. The subsumption provided automatically by the network may allow us to perform more efficient pattern matches. Also, the network will maintain a rule hierarchy within the heretofore unstructured rule set. This has potential as an aid to knowledge base maintenance and debugging.

Summary

The domain of financial marketing has proved to be a complex one for which the exclusive use of either rules or an inheritance network has been demonstrated to be inadequate. We find that our integrated use of rule-based inference and an inheritance network provides substantial facilities for flexible problem solving. The rules provide a efficient pattern match capability and inference cycle for control. The inheritance network provides a convenient way to represent the conceptual structure of the domain. The unique features of our inheritance network and rule language work together to provide a powerful representation and problem solving facility.

Acknowledgments

We are indebted to numerous people from the marketing divisions of IBM for providing expertise on this project. Chidanand Apté, Jim Griesmer, Joe Parzych, and Se June Hong provided many useful comments on earlier drafts of this paper.

References

1. Mays, E., Apté, C., Griesmer, J., and Kastner, J., Organizing Knowledge in a Complex Financial Domain, *IEEE Expert*, Vol. 2, No. 3, pp. 61-70, Fall 1987.

2. Schor, M., Daly, T., Lee, H. S., Tibbitts, B., Advances in RETE Pattern Matching, *Proceedings of the 1986 National Conference on Artificial Intelligence*, pp. 226-232, Philadelphia, Pa., Aug. 1986.

3. Bobrow, D. and Winograd T., An Overview of KRL, a Knowledge Representation Language, *Cognitive Science* 1, 1977, pp. 3-46. (Reprinted in *Readings in Knowledge Representation*, edited by R. J. Brachman and H. J. Levesque, Morgan Kaufman, Los Altos, Ca, 1985.)

4. Brachman, R. J. and Schmolze J. G., An Overview of the KL-One Knowledge Representation system, *Cognitive Science* 9, 1985, pp. 171-216.

5. Fikes, R. and Kehler, T., The Role of Frame-based Representation in Reasoning, *Communications of the ACM* 28, Sept. 1985, pp. 904-920.

6. Forgy, C. L., Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem, *Artificial Intelligence* 19, Sept. 1982, pp. 17-37.

7. Kastner, J., Apté, C., Griesmer, J., Hong, S. J., Karnaugh, M. , Mays, E., and Tozawa, Y., A Knowledge-Based Consultant for Financial Marketing, *AI Magazine*, Vol. 7, No. 5, pp. 71-79, Winter 1987.