Generation, Refinement, and Extension of Explanation for Cooperative Information Systems

Michael J. Minock minock@cs.ucla.edu

Wesley W. Chu wwc@cs.ucla.edu

December 1995

Computer Science Department,
University of California at Los Angeles

1.0 Introduction

Systems must become better at explaining the operations they offer, the actions they take, and the results they yield. Currently many systems do this to one degree or another. Word processors and spreadsheets offer simple, self-explanatory operations and show direct results immediately. Information systems offer custom query forms and data-entry screens and simple answer set presentations or reports. These systems have succeeded in their communication responsibilities by simplifying the set of operations they offer and by presenting the application state graphically in accordance with familiar, intuitive metaphors. Yet in many instances we wish to grant systems broader, more complex responsibilities. Simple communication techniques are insufficient in these cases. If required, such systems must be able to explain themselves [7][11][10][6].

One area in which explanation technology is required is in Cooperative Information Systems[5]. Traditionally database systems accept precise query specification (SQL, Datalog, etc.) and return exact answer sets. In Cooperative Information Systems people pose imprecise queries and receive approximations when answer sets are non-existent and summaries when answer sets are large. Yet these cooperative answers should be accompanied by an explanation of how they were derived. This work addresses the use of explanation technology in the cooperative information system CoBase[1][2][3][8]. CoBase provides approximate and associated answers to a user's relational and object-oriented queries. CoBase's explanation system provides multimedia, interactive, user-sensitive and context-dependent descriptions and explanations. Specifically the system describes CoBase queries, knowledge, and answers and explains the relaxation and association processes as well as the quality of answers.

1.1 Example

CoBase query relaxation will serve as the example here. Query relaxation is the process of weakening query constraints to increase the set of answers that satisfy the query. This is useful when a query does not return any answers, or the user is interested in exploring the semantic neighbors of the answers returned. Suppose the user poses a query seeking airports in Bizerte, Tunisia where runway lengths are greater than 5800'. It happens that there are no such airports, so CoBase relaxes the user query from the location Bizerte to the region of Northeast Tunisia. After doing so CoBase finds a neighboring airport in the Northeast Tunisian city, Djedeida. Figure 1.1 appears.

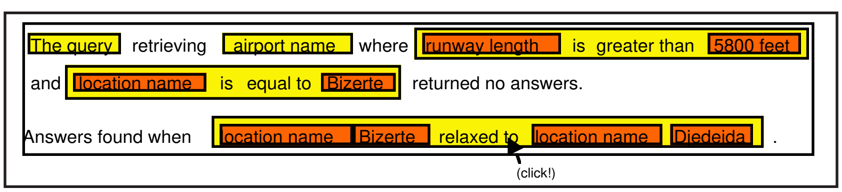

Figure 1.1: Message sent after user query was processed.

The explanation is brief. If the user requires more detail they interact with this initial explanation, obtaining further description. Sensitive text fragments are stacked hierarchically, letting the users interact with a single word, phrase, sentence, or paragraph. When the users "clicks" as above, figure 1.2 appears.

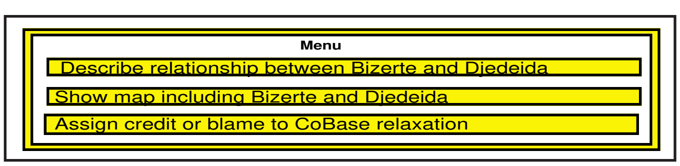

Figure 1.2: A menu presented to the user after the "click" of figure 1.1.

If the first option is clicked then figure 1.3 appears, and if the second option is clicked figure 1.4 appears. If the third option is clicked then the user is given the option of providing positive or negative feedback to refine CoBase's relaxation strategy. Such feedback is then absorbed into the CoBase relaxation control knowledge.

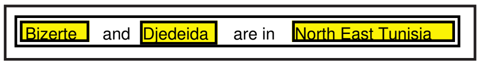

Figure 1.3

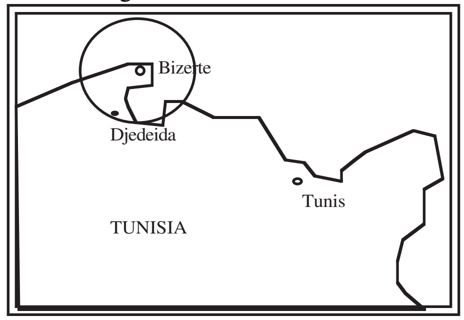

Figure 1.4

Although not illustrated in this example, explanation is user and context sensitive. Based on the user and context, explanations should occur at appropriate times during processing. One extreme is the system running automatically, only summarizing its work once it has completed. Another extreme is the system running in detail, explaining each action, giving the user the ability to monitor progress. Explanations should also be tailored to a user's understanding of the system.

Finally explanations must be extendible. CoBase is designed as a general system, capable of being applied to a wide range of domains. As such, any explanation system tied to CoBase must be domain-independent. Though a domain-independent explanation may suffice, users prefer to have explanations as domain specific as possible. Hence it is important to have a facility that allows initially general CoBase explanations to be mapped to more domain specific explanations. For example the explanation in figure 1.1 could be better tailored to the transportation domain as in figure 1.5. Through a series of editing operations, the user transforms the initial, domain independent explanation into a domain specific form. Henceforth the explanation system will generate this explanation when working in the transportation domain.

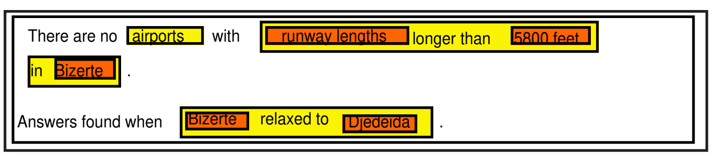

Figure 1.5: User improvement of explanation in figure 1.1.

1.2 Problem and Objectives

We have developed an explanation system[9] to provide the above capabilities. In addition to addressing the role of explanation in the Cooperative Information System CoBase, this work also addresses the more general problem of how to interactively extend machine-generated explanations and refine their patterns of expression. The position here is that it is unrealistic to expect all patterns of explanation to be present a priori. Initial explanations are expected to be inadequate. Through a set of editing and feedback operations, knowledgeable users extend and refine machine-generated explanations, enabling the explanation system to be more articulate in the future.

2.0 System

The explanation system is *capable* of providing multimedia, interactive, user-sensitive and context-dependent explanations of CoBase cooperative operations. The explanation system *requires* access to queries, TAHs[1], execution traces,

---

[1]TAHs (Type Abstraction Hierarchies) are the principle knowledge structure in CoBase used to guide the query transformation process.

and answers (Figure 2.1). A client sends the explanation system a simple *explanation request* on pieces of this information (e.g. describe query, explain answers) and receives back an *explanation reply* consisting of natural language and recommended visualizations that solve the request. The client presents the explanation on their GUI and a simple protocol enables the user to interact with initial explanations, causing the explanation system to generate follow up explanations. The explanation system maintains a model of query processing, consisting of concepts and classification rules, by which it interprets CoBase queries, TAHs, execution traces and answers. The explanation system applies generation rules to produce explanation replies from explanation requests.

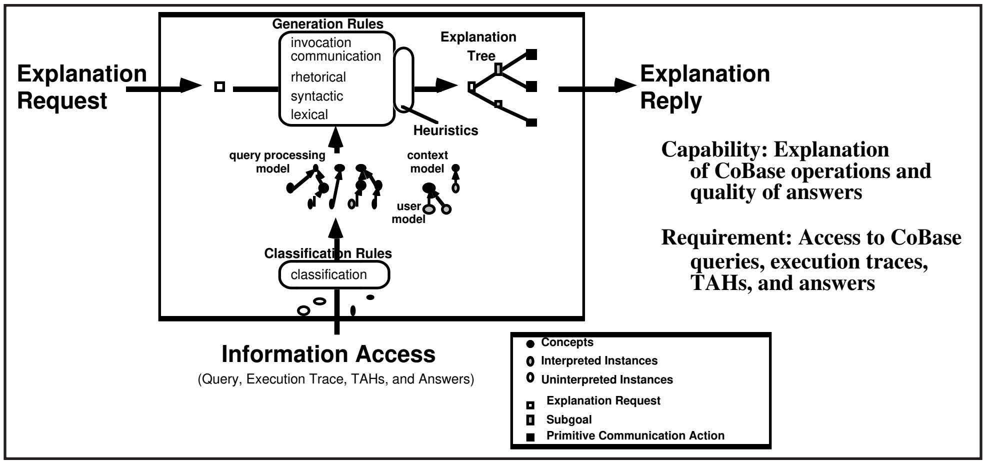

Figure 2.1: explanation system architecture.

The explanation system represents information such as queries, execution traces, type abstraction hierarchies, and answers in a specific type of semantic network (graph) offering several advantages over standard semantic networks for explanation. Most importantly it allows for abstraction on the labels linking nodes (instances, values, or concepts), enabling specific to generic access of the graph. The representation also enables the direct expression of one-to-many, many-to-one, and many-to-many relationships. Concepts in this graph (e.g. Query, Relax-Query-Action, etc.) include those of the query processing model . As the CoBase system executes, it sends a stream of information that is instantiated in this graph. Classification rules in the query processing model interpret this information as it added to the graph. In addition the graph contains a user and context-model, enabling the generation of user-sensitive, context-dependent explanations.

The explanation system accepts an explanation request on a node (or set of nodes) in the graph and produces an explanation tree. This tree is created via the depth-first application of generation rules toward solving the initial explanation request. The explanation request is at the root of the explanation tree, while the text and recommended visualizations of the explanation are leaves in the tree. Intermediate nodes are the subgoals that were carried out to achieve the root explanation request. Generation rules consist of a goal, a set of constraints, and a set of actions. If the explanation request (or an intermediate subgoal) is matched by a generation rule goal, and if all of the rule's constraints are met, then the rule's actions are performed. These actions may either be subgoals that are in turn solved by application of generation rules or may be primitive communication actions (e.g. text or a visualization call). Rule constraints and actions access information in the graph as the explanation tree is being synthesized.

There are five types of generation rules: invocation, communication, rhetorical, syntactic and lexical. Invocation rules determine what should be explained under various user models and contexts. Communication rules determine what information will be expressed. Rhetorical rules determine the manners in which information is expressed. Syntactic rules constrain text to valid English. Lexical rules determine the names that will be given to CoBase actions and objects.

Invocation and communication rules are typically the rules that match explanation requests while rhetorical, syntactic, and lexical rule typically expand intermediate nodes and produce primitive communication actions.

3.0 Discussion

Different users require different types of explanation under different contexts. Since alternate generation rules vary in which aspects of CoBase's process or results should be described and to what depth, this problem reduces to the proper selection generation rules given the current user and context. This may be handled directly by conditioning generation rules on the user and context models. However it is assumed that this treatment will not be complete and that, in the field, users must be able to interactively indicate their preferences under various contexts. The approach we are pursuing is to absorb user feedback on explanations into the heuristic knowledge that determines which generation rule to apply among the set of rules that match to expand a goal. The particular heuristic structure we use is our *generation rule override graph.* The generation rule override graph expresses precedence relationships among generation rules given the user, the context, and the sequence of generation rules firing immediately prior to rule selection. Initially empty, this graph is populated with edges as the user provides feedback on explanations. We are currently optimizing algorithms on the generation rule override graph and are exploring how to detect and resolve potential inconsistencies in the user's feedback.

New generation rules may need to be added to the explanation system as CoBase is applied to new domains. To our knowledge all current generation explanation systems require that new generation rules be manually written and added to their rulebases. We seek to make this process interactive. Through a sequence of editing operations (expand, delete, change, and move), users transform initially general explanations into precise, domain specific forms. In transforming the visible explanation, users transform and mark the explanation tree of the initial explanation. Once editing is complete, the transformed, marked explanation tree contains the new domain specific explanation. This transformed explanation tree is traversed and when marked nodes are encountered new generation rules are induced from the original generation rules that expanded the marked node. These new generation rules are constrained to fire only when the current context and user model are active and override the rules from which they were induced. By this process the generation rulebase is extended interactively.

3.0 Conclusions

The explanation system is implemented in C++ and CLIPS. There are approximately 30 concepts and 20 classification rules in the query processing model. There are approximately 80 general generation rules and 40 domain specific rules for an electronic warfare domain and 30 rules for a transportation planning domain. Elapsed times depend on access cost to such information and the computational complexity of synthesizing the explanation tree. The dominant cost is the synthesis of explanation trees, particularly the cost of matching explanation rules to expand nodes. A RETE [4] pattern-matcher performs these matches. The explanation system can usually produce explanations in under a second on a Sun Sparc 10 workstation. The quality of explanations (with respect to completeness, correctness, and precision) depends on breadth of access to CoBase queries, execution traces, TAHs, and answers, and also on the effort expended in populating and refining the explanation rulebase.

References

[1] Chu, Wesley W. and Chen, Q. Neighborhood and Associative Query Answering. *Journal of Intelligent Information Systems*, 1(3/4), 1992.

[2] Chu, Wesley W. and Chiang, Kourong. Abstraction of High Level Concepts from Numerical Values in Databases. *Proceedings of the AAAI workshop on Knowledge Discovery in Databases,* July 1994.

[3] Chu, W., Yang, H., Chiang K., Minock, M., Chow, G. and Larson, C. CoBase: A Scalable and Extensible Cooperative Information System. Submitted to *Journal of Intelligent Information Systems*, September 1995.

[4] Forgy, C.L. RETE: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem*, Artificial Intelligence* 19, 1982, 17-38.

[5] Gaassterland, T., Godfrey, P. , and Minker, J. An Overview of Cooperative Answering. *Journal of Intelligent Information Systems*, 1(2):127-157, 1992. Invited paper.

[6] Kantrowitz, Mark and Bates, Joseph. 1992. *Integrated Natural Language Generation Systems* Aspects of Automated Natural Language Generation, Springer-Verlag.

[7] McKeown, Kathleen R. Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text. *Cambridge University Press*, Cambridge, England, 1985.

[8] Merzbacher, Matthew and Chu, Wesley W. Pattern-based clustering for database attribute values*, Proceedings of the AAAI workshop on Knowledge Discovery in Databases*, 1993.

[9] Minock, Michael J. and Chu, Wesley W. Interactive Explanation for Cooperative Information Systems. *UCLA Computer Science Department Technical Report*, 1995.

[10] Moore, Johanna D. and Paris, Cecile L. 1989. Planning Text for Advisory Dialogue. *In Proceeding of the Twenty-Seventh Annual Meeting of the Association for Computational Linguistics*, Vancouver, British Columbia.

[11] Swartout, William R., Paris, Cecile L., and Moore, Johanna D. Design For Explainable Expert Systems. *IEEE Expert*, Volume 6, Number 3, pages 58-64.