MACRON: An Architecture for Multi-agent Cooperative Information Gathering

Keith Decker, Victor Lesser, M. V. Nagendra Prasad and Thomas Wagner*
Department of Computer Science
University of Massachusetts
Amherst, MA 01003, USA.
{decker,lesser,nagendra,wagner}@cs.umass.edu

September 21, 1995

Abstract

The complexity of the modern information carrying landscape requires a sophisticated view in which information is *acquired* rather than simply retrieved; where the process must be dynamic, incremental, and constrained by resource limitations. In this paper, we describe a multi-agent architecture called MACRON based on the Cooperative Information Gathering (CIG) paradigm designed specifically for such complex information gathering environments. Top level user queries drive the creation of partially elaborated information gathering plans, resulting in the employment of multiple cooperative agents for the purpose of achieving goals and subgoals within those plans. Agents in MACRON incorporate capabilities to exploit subproblem interdependencies, manage uncertainty inherent in multi-agent search, intelligently trade-off solution quality for resource limitations and exploit or avoid redundancy as needed.

1 Introduction

The complexity of the modern information carrying landscape requires a sophisticated view in which information is *acquired* rather than simply retrieved; where the process must be dynamic, incremental, and constrained by resource limitations. Information Gathering (IG) is an activity involving pro-active acquisition of information, from possibly heterogeneous sources, in response to a complex query that may require the system to possess capabilities such as reasoning, representation, and inferencing. The set of data that represents the best response to a user's query may be the aggregation of data acquired from distributed, heterogeneous information sources. In addition to the complexity of query specification, control

*This material is based upon work supported by the NSF Center for Intelligent Information Retrieval, ONR contract N00014-92-J-1450, ARPA under ONR contract N00014-92-J-1698. The content of the information does not necessarily reflect the position or the policy of the Government and no official endorsement should be inferred.

of the acquisition process may itself be complex and dynamic in IG systems. Traditional Information Retrieval (IR) is a limited sub-case of such information gathering systems in which queries generally map onto static, pre-specified retrieval plans.

We recently presented a model of *Cooperative Information Gathering* (CIG)[11] designed specifically for such complex information gathering environments. In this paper, we describe an architecture called MACRON (**M**ulti-agent **A**rchitecture for **C**ooperative **R**etrieval **ON**line), based on the CIG paradigm for multi-agent information gathering activities. Top level user queries drive the creation of partially elaborated information gathering plans, resulting in the employment of multiple cooperative agents for the purpose of achieving goals and subgoals within those plans. The agents use existing information resources available on the Internet such as newsgroups, archives, magazine databases, and corporate WWW sites. Such a process can be described as a "distributed problem solving" approach where the agents are autonomous, socially situated, and can react to a dynamically changing environment. For example, agents may explicitly cooperate to achieve their goals. Cooperation between agents implies management of interdependencies between their activities so as to integrate and evolve consistent clusters of high quality information from distributed heterogeneous sources. Agents may share information with one another that affects what plans they execute to achieve their assigned sub-goals, what order they execute actions, and when they execute them. Another feature is that the agents are time-aware and are free to develop and follow different information gathering plans that depend on the amount of time they have to produce an answer. Eventually, some agents will use technologies like natural language form-filling programs and the INQUERY indexing scheme to make decisions about how to focus their search without constantly requiring detailed user feedback.

A multi-agent approach to information gathering is a promising one for a variety of reasons.

- Multiple agents offer concurrency which is a big win in time constrained situations or when the search space is very large as in the case of networked information gathering. The query plans can often be decomposed into relatively independent sub-plans with a few interdependencies. An agent executing a sub-plan functions relatively autonomously but needs to coordinate with others where interdependencies exist.

- When a system is dealing with enormous quantities of data, distributed computation at the sites where the data resides may often be the more viable approach compared to migrating data to a centralized processing location. Instead of gathering data dispersed across networked information servers at a centralized site and then evolving a coherent response to a query, agents can reside at the sources and perform distributed coordinated retrieval to prune their data space and convey a much smaller subset to the query system for further processing.

- Agent-based architectures offer modularity, robustness, separation of concerns, and other advantages of a distributed system. Information agents can be constructed and maintained separately. An agent can build upon other information agents to provide an abstraction of these information sources. In addition, passive data sources like databases can be transformed into information agents by wrapping them with intelligent interfaces[9].

The rest of the paper is organized as follows. Section 2 briefly introduces the CIG paradigm with an example. Section 3 introduces the MACRON architecture and discusses it in detail. The following section discusses the status of implementation and our future plans. The final two sections discuss related work and conclude.

2 Cooperative Information Gathering (CIG)

CIG model[11] views information gathering as a distributed problem solving activity characterized by the existence of interdependencies between subproblems leading to a need for the agents to cooperate extensively during problem solving. Lesser [10] presents the functionally accurate, cooperative (FA/C) paradigm as an approach to distributed problem solving. In FA/C systems, various soft and hard constraining *goal/task interrelationships* among subproblems motivate agents to augment their local information with information about global problem solving activity in order to enhance the efficiency of the ongoing problem solving process. Once uncovered via communication of problem solving activities, such as receiving partial results or meta-information, these interdependencies can be exploited in a variety of ways to improve problem solving both locally and globally. In a CIG task, potentially useful constraints may exist between different pieces of information. The discovery and exploitation of such constraints is necessarily a dynamic and incremental process that occurs during problem-solving and entails communication of partial results among agents in a timely and selective manner, to augment each agent's local view with a more global view. Given the incomplete nature of the local views of the individual agents, another important aspect of FA/C is the explicit recognition of the role of solution and control uncertainty. Coupled with the fact that resources and time for conducting a search are limited in real-life problems, this leads to the notion of *satisficing search*. Another aspect of FA/C is the explicit recognition and exploitation or avoidance of *redundancy*, leading to increased robustness or decreased resource demands depending on the context and the structure of the domain.

The organization of information on the Internet exhibits many of the characteristics of real world problems that originally prompted the evolution of the FA/C approach. Most information on the Internet, be it the new version of a shareware program or the review of a certain software package, is in fact often available in many physical locations. Some of these locations are mirrors, presumably up-to-date copies of certain well-known information collections, but others are different (possibly mirrored) collections that partially overlap in content and may be organized differently. Network access to any particular location may change minute-to-minute depending on network loads and hardware. There is a need to provide seemingly replicated access to known information sources without excessive redundant retrieval of information by multiple agents. Information exhibits "connectivity" in the sense that a piece of information contains references to other relevant information. For example, a web page might be retrieved that contains links to new pages that have a high probability of containing useful information but that are not a part of the current retrieval plan. Partial search results may be presented to the user in order to get feedback to focus the remaining search. The search agents can redirect their processing resources and information gathering plans as information is retrieved during query processing. For example, if the user is looking for information on color printers, the system may discover that there are probably too many models and would then present the user with a list of features (such as price or print quality) that the user can interact with to narrow the search.

Figure 1 shows an example that highlights the same issues in the domain of computer product reviews[4]). For a given query, there may be many sources of relevant information. Product reviews often exist on-line, or may obtained from publishers for a fee in paper or electronic format. Relevant reviews may be found on-line in the review section of the TidBits newsletter, in the Info-Mac archives, or in discussions about the product in Usenet news groups. The query may be satisfied by dispatching agents to locate the required review and then retrieve it. Each agent may employ different access methods (such as WAIS, FTP, HTTP, telnet, etc.), and the access methods may have recourse to the same information at a variety of physical locations (such as the main TidBits archive ftp.tidbits.com or its various mirrors).

2.1 Subproblem Interaction

Various kinds of goal interrelationships like facilitates, enables, overlaps, and subsumes that exist between subproblems can be detected and exploited in a variety of ways [5, 6]. For example, uncertainty that may arise from incomplete local information can be reduced through detection and subsequent exploitation of overlaps and subsumes interrelationships. However, the process of detecting and exploiting goal interrelationships involves providing the agent with a more complete global view, and that entails communication costs. Hence an agent should communicate only relevant portions of its local view of the problem solving process to help form a more coherent view of the emerging global problem solving process in other agents. Partial solutions and meta-information received from other agents may facilitate (i.e. serve to focus or constrain or lend support to) a local solution or may point to an inconsistency in an agent's local processing or detect redundancies. The end effect is "better" or higher quality global solutions and reduced computational requirements.

In our example, locating a paper review "enables" its retrieval, i.e. paper reviews may be obtained by first finding a citation, and then either finding the actual article or obtaining it from the publisher. Finding a citation via Uncover "facilitates" the goal of getting the article faxed to the user. An overlaps interrelationship exists between goal "Get from Seller" and goal "Use Uncover". This is due to the fact that once the seller's archive offers a particular citation, the search for that same citation at the Uncover database can be avoided. Another source of information that is not exploited in the example above is the increasingly popular World Wide Web (WWW). We can think of the web of citations or the web of hyper-text links associated with a document as a web of consistency constraints. That is, documents linked in this way may contain related information. For example, during the process of retrieval of product review discussed above via WWW server, two sites may quote different prices for the product. Product review information at an FTP site accessed via WWW may contain outdated prices, whereas a link to the seller database in a html document may in fact contain the latest prices. When inconsistencies are uncovered, agents need to work to resolve the associated uncertainty so that a cluster of consistent documents containing "correct" information is presented to the user. In this case, the agents may choose to use the seller database information and override other sources for the price information.

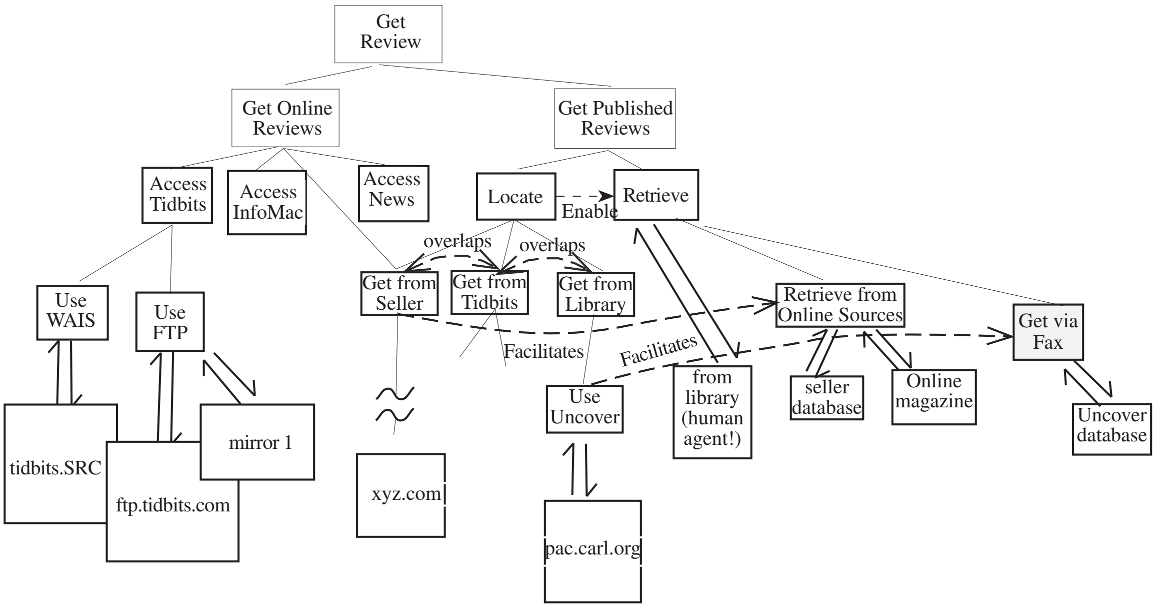

Figure 1: A goal tree for retrieving Macintosh related product reviews

2.2 Satisficing

Although the amount of information available on the Internet is seemingly boundless, the resources like time, money, and computational resources available to search that information typically are not. Rather than being able to develop an exhaustively complete and accurate response to a query, intermediate results from disparate sources must be pieced together to form consistent islands of information that can be incrementally refined to form a more accurate solution depending on the extent of available resources and time. That is, the information gathering process must be *satisficing* along various dimensions like precision, quality, etc. If communication is slow, we may access nearby data with low expected quality first, rather than trying distant data sources of higher quality that may require more time than is available. When more time is allocated to the search process, the scope of the search can be broadened to include higher quality sources, while retaining some amount of effort on inexpensive low quality sources.

2.3 Redundancy

Redundancy in distributed search, either in the form of replication of data at multiple sites or the possibility of deriving the same conclusion from different sets of data, raises a host of issues. Advantages of redundancy include increased robustness of the system in environments with failure-prone components and increased flexibility in responses. Redundancy can play a role in the reduction of uncertainty when dealing with erroneous or incomplete information. On the other hand, redundancy has the disadvantage of increased resource usage and possibly increased total processing times. For example, the Internet may contain "mirror" sites for certain data repositories or it may contain redundant data from different sources for the same task. Data from different sources may be of different quality or may be differently organized.

A particular task could possibly do with low quality data which perhaps could be locally acquired. Thus, recognizing the role of redundant data and computation could be important for exploiting the possibilities such redundancy offers in a CIG system. Redundancy could be permitted if the control costs outweigh the benefits of avoiding it. Alternatively, if we are dealing with faulty systems or poor quality data, redundancy could help resolve the uncertainty in the retrieved data by providing additional constraints.

3 MACRON: An Architecture for CIG

MACRON is a multi-agent architecture that aims to implement a CIG system incorporating capabilities to exploit subproblem interdependencies, manage uncertainty inherent in multi-agent search, intelligently trade-off solution quality for resource limitations and exploit or avoid redundancy as needed. We see MACRON as consisting of four major parts: 1) Organizational architecture 2) Agent architecture 3) Low level retrieval 4) User interface. This division is an artificial one, and has been adopted here purely for pedagogical purposes.

3.1 Organizational Architecture

From an organizational point of view, the agents form a matrix organization consisting of both functional and query-answering units as shown in Figure 2. A functional unit provides access to some particular type of information resource such as current and archived news feeds, or magazine databases. Each functional unit has a functional manager (FM) agent who is responsible for task assignments to agents within the unit. Functional units may comprise relatively undifferentiated agents that have similar knowledge and capabilities, or differentiated agents with fairly focussed skills. A FM agent has an abstract view of the overall state of its functional unit and hence can perform task allotment in an intelligent manner taking resource constraints into consideration[1]. A query-answering unit comprises a query-manager agent and a set of agents drawn from the various functional units to respond to a particular user's query. The query manager is responsible for developing an initial high-level information gathering plan, recruiting agents from the necessary functional units (via the functional unit managers), and monitoring the plan's execution (including keeping the user up-to-date and able to steer the plan as it is carried out). The user interacts with a query manager via a web browser and HTML forms. Agents interact with one another using KQML (Knowledge Query and Manipulation Language, part of the ARPA Knowledge Sharing Effort) and a task structure specification language based on the TÆMS (Task Analysis, Environment Modeling, and Simulation) representation framework. TAEMS represents the structure of tasks (like information gathering), including the hard and soft relationships between tasks, for the purpose of coordinating and scheduling agent actions. For example, in Figure 1, the query agent can expand the initial "Get Review" goal to "Get Online Reviews" and "Get Published Reviews". It then recognizes that it has to communicate with FM agents for "Online Reviews" and "Published Reviews". These managers in turn allocate the goals "Get Online Reviews" and "Get Published Reviews" to the agents in their

[1]The same allocation can be achieved in a distributed manner by the agents involved through coordinated distributed scheduling.

respective functional units. Say in this example, each goal is allocated to one agent. The agents expand their goals to reach the lowest level primitive tasks like Use Wais, Use FTP, Retrieve Online Magazine(x) etc.

The organization chart manager (OCM) is an organizational memory store that allows an agent to get information about which other agents to communicate with when shipping a task or sending coordination information. New agents and new functional units can be added to a running system by adding them to the organizational chart manager. For example, the query agent checks with OCM to find the address of the functional unit manager that can achieve “Get Online Reviews”.

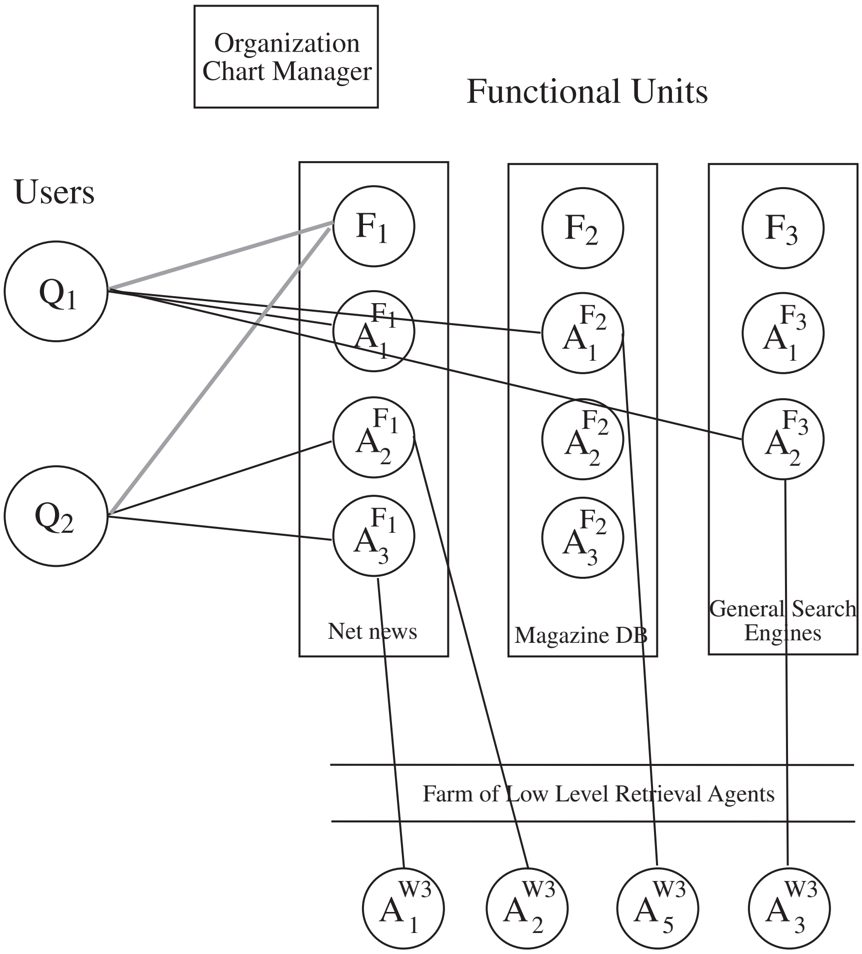

Figure 2: Organizational Architecture for MACRON

3.2 Agent Architecture

Figure 3 shows the architecture of an individual agent consisting of several components: a task structure database that contains the agent’s current view of its tasks and their relation to other agent’s tasks; a coordination module that reacts to certain patterns in the task structure database and causes the creation of new local scheduling constraints, called commitments, which can be communicated to other agents; a planning module that can elaborate the task structure representation; a local scheduler that takes as input the current task structure, local commitments, and non-local commitments (from other agents) and produces several possible schedules (what actions to take, in what order, and at what point in time); and an execution monitor that oversees action execution to anticipate problems and slipped deadlines. This architecture is based on the GPGP (Generalized Partial Global Planning) architecture described in detail in [5, 6]. For example, upon receiving its task "Get Online Reviews", the "Online Reviews" agent calls a planner to expand the task into a query plan consisting of primitive actions like "Use FTP" and "Use Uncover". A real-time scheduler is invoked in conjunction with coordination mechanisms to enable the agent to form a schedule that best serves the interest of the global query plan. For example, the scheduler realizes that "Use Uncover" facilitates "Get via Fax" task in the "Published Reviews" agent and hence schedules it in the earliest possible slot in order to be able to communicate relevant partial results to the other agent. The execution monitor keeps track of the progress in the plan execution and initiates appropriate measures for re-planning or rescheduling as needed.

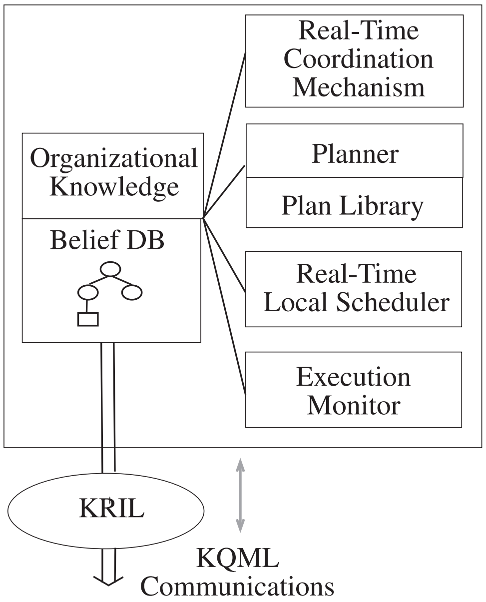

Figure 3: DECAF: Distributed Environment Centered Agent Framework

3.3 Low-level Retrieval Agents

The low-level retrieval agents are the system's interface to the World-Wide-Web. The agents, referred to as "getters", are responsible for retrieving and processing URLs for the higher-level planning agents and interacting with remote search engines and databases. Each of the primitive tasks in a schedule developed by a higher-level agent results in the recruitment of one or more of these low-level retrieval agents to achieve the task. Getter services include MIME type retrieval via URL specification, HTML 2.0 parsing, keyword match and selection of HTML or text documents, selection and extraction of embedded URLs, and the ability to interact with HTML forms. Like the other agents in the MACRON architecture, getters communicate using KQML. Getters use domain specific knowledge about particular sites and fill-in HTML forms to determine how to interact with remote search engines and query forms and how to properly process their output. Through these services the getters provide a necessary abstraction of the WWW and its heterogeneous information resources to the rest of the agent system.

3.4 User Interface

In developing the user interface, we vested the human users with the ultimate authority to override or redirect the search process as they deem appropriate. However, such a philosophy gives rise to the necessity for careful design of the user interface. The time-line of a complex multi-agent search process has to be displayed to the user in a cognitively compatible manner and his or her feedback has to be mapped back to the distributed search control process.

The user interface for MACRON displays the evolving task structures and the user can choose a task to get a more detailed view of its status. The user is permitted to make changes to the modifiable parameters of the task and submit the changes which are then mapped back to the agents to be assimilated into their search control.

4 Status of Implementation

MACRON is in its early stages of development. The basic agent communication infrastructure is the KQML-based Lockheed KAPI integrated with Harlequin Lispworks. We have a rudimentary multi-agent information gathering system[1] with a Mosaic forms interface that receives a query from the user and invokes a query agent. The query agent invokes a hierarchical case-based planner that instantiates an information gathering plan and invokes the appropriate low level agents. These agents in turn communicate with the actual information sources. As of now we have low level agents that can communicate with ten different information sources (Djanews,Macinfo, INQUERY etc) available on the World Wide Web. The planner agent was setup to answer queries on Macintosh products.

5 Conclusion

Information Gathering systems proposed in the literature typically either do not fully exploit the potential of knowledge-intensive methods for the task [2, 3, 7] or tend toward distributed processing, failing to exploit the dependencies between agents working on different aspects of an information gathering task[12]. The system by Knoblock et al.[8] is an exception in the sense that it is a knowledge-intensive and has some notion of exploiting dependencies. However, their system has no notion of scheduling to maximize quality given deadlines and cost limits. MACRON is an attempt to overcome these shortcomings. MACRON is based on our strong belief that cooperating to enhance efficiency of a resource-limited information acquisition process in complex, heterogeneous and unstructured data environments can be viewed as a distributed problem-solving task within the FA/C paradigm.

References

[1] CIG Searchbots . `http://dis.cs.umass.edu/~wagner/searchbots/searchbots.html`.

[2] M. C. Bowman, P. B. Danzig, U. Manber, and M. F. Schwartz. Scalable internet resource discovery: Research problems and approaches. *Communications of the ACM*, 1994.

[3] J. Callan. Personal communication, July 1994.

[4] Keith S. Decker. *Environment Centered Analysis and Design of Coordination Mechanisms.* PhD thesis, University of Massachusetts, 1994.

[5] Keith S. Decker and Victor R. Lesser. Generalizing the partial global planning algorithm. *International Journal of Intelligent and Cooperative Information Systems*, 1(2):319–346, jun 1992.

[6] Keith S. Decker and Victor R. Lesser. Designing a family of coordination algorithms. In *Proceedings of the First International Conference on Multi-Agent Systems*, pages 73–80, San Francisco, June 1995. AAAI Press. Longer version available as UMass CS-TR 94–14.

[7] M. Huhns, U. Mukhopadhyay, L. M. Stephens, and R. Bonnell. DAI for document retrieval: The MINDS project. In *Distributed Artificial Intelligence.*

[8] C. A. Knoblock and Y. Arens. An architecture for information retrieval agents. In *Working Notes of the AAAI Spring Symposium on Software Agents.*

[9] C. A. Knoblock, Y. Arens, and C. Hsu. Cooperating agents for information retrieval. In *Proceedings of the 2nd International Conference on Cooperating Information Systems*, 1994.

[10] V. R. Lesser. A retrospective view of FA/C distributed problem solving. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6):1347–1363, November 1991.

[11] T. Oates, M. V. Nagendra Prasad, and V. R. Lesser. Cooperative Information Gathering: A Distributed Problem Solving Approach. Computer Science Technical Report 94–66, University of Massachusetts, 1994.

[12] E. M. Vorhees. Software agents for information retrieval. In *Working Notes of the AAAI Spring Symposium on Software Agents.*