Functionally Accurate, Cooperative Distributed Systems

Victor R. Lesser and Daniel D. Corkill

Appeared in *IEEE Transactions on Systems, Man, and Cybernetics,* SMC-11(1):81–96, January 1981.

Abstract

A new approach for structuring distributed processing systems, called functionally accurate, cooperative (FA/C), is proposed. The approach differs from conventional ones in its emphasis on handling distribution-caused uncertainty and errors as an integral part of the network problem-solving process. In this approach nodes cooperatively problem solve by exchanging partial tentative results (at various levels of abstraction) within the context of common goals. The approach is especially suited to applications in which the data necessary to achieve a solution cannot be partitioned in such a way that a node can complete a task without seeing the intermediate state of task processing at other nodes. Much of the inspiration for the FA/C approach comes from the mechanisms used in knowledge-based artificial intelligence (AI) systems for resolving uncertainty caused by noisy input data and the use of approximate knowledge. The appropriateness of the FA/C approach is explored in three application domains: distributed interpretation, distributed network traffic-light control, and distributed planning. Additionally, the relationship between the approach and the structure of management organizations is developed. Finally, a number of current research directions necessary to more fully develop the FA/C approach are outlined. These research directions include distributed search, the integration of implicit and explicit forms of control, and distributed planning and organizational self-design.

Manuscript received November 30, 1979; revised September 10, 1980. This work was supported, in part, by the National Science Foundation under Grant MCS78-04212, and in part by the Office of Naval Research under Grant N00014-79-C-0439. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official opinion or policy of the University of Massachusetts or any agency of the U.S. Government.

I. Introduction

Recent developments in microprocessor technology [62] and network technology [8, 33] have lowered the cost of processors and communication to a level where distributed processing is now practical. The potential advantages of a distributed processing approach over a centralized approach include [39]:

- *increased reliability and flexibility*—achieved through redundancy in communication paths and processing nodes and through modularity of design (which permits incremental addition of new processing nodes and communication paths)
- *enhanced real-time response*—achieved through parallelism and through the placement of processing nodes near sensing devices and devices to be controlled
- *lower communication costs*—achieved by abstracting (preprocessing) data for transmission (lowering communication bandwidth requirements) and by placing processing nodes near the data (reducing the distance over which the data must be transmitted)
- *lower processing costs*—achieved through the use of cheaper, less complex processors that can be mass produced and through load sharing (allowing relatively idle processing nodes to handle some of the work of a busy processing node)
- *reduced software complexity*—achieved by decomposing the problem-solving task into subtasks, each more specialized than the overall task; the result of this decomposition is reduced software complexity at each processing node (which performs a small number of subtasks) as compared to software performing the complete task.

These potential advantages have yet to be exploited in a wide range of application areas. Only in the areas of process control [13, 30, 54] and distributed data bases [1, 45] have some of the promises of distributed processing been realized. Applications in these areas are characterized by task decompositions in which the data can be partitioned in such a way that each subtask can usually be performed *completely* by a single node—without the need for the node to see the intermediate states of processing at other nodes.

The authors are with the Computer and Information Science Department, University of Massachusetts, Amherst, MA 01003.

A number of additional applications that appear naturally suited to distributed implementation (such as sensor networks, automotive and air-traffic control, power-network grids, and tasks involving mobile robots) do not have the task decomposition characteristics of conventional distributed processing applications and therefore appear ill-suited to conventional approaches. In these applications, the data necessary to achieve a solution *cannot* be partitioned in such a way that a node can complete a subtask without seeing the intermediate state of processing at other nodes.

An example of this type of application is distributed vehicle monitoring. Vehicle monitoring is the task of generating a dynamic area-wide map of vehicles moving through the monitored area. In one distributed version of this task [23, 34, 38, 58], processing nodes, with their associated acoustic sensors (of limited range and accuracy) are geographically distributed over the area to be monitored. Each processing node can communicate with other nearby nodes over a packet-radio communication network [31]. Because acoustic sensors characteristically produce a significant amount of error, the purely localized processing of sensory data would result in the "identification" of nonexistent vehicles, the missed detection of actual vehicles, and the incorrect location and identification of actual vehicles. In this application, the amount of communication required to redistribute the raw sensory data necessary for correct localized processing would be significant.

An alternative approach for resolving these errors is for processing nodes to interact in a highly cooperative way, exchanging tentative partial results with one another. For example, each node's tentative vehicle identifications can be used to indicate to other nodes the areas in which vehicles are more likely to be found and the details (vehicle type, rough location, speed, etc.) of probable vehicles. In addition, consistencies between these tentative identifications serve to reinforce confidence in each node's identifications. Such cooperation is not only appropriate for vehicle identification, but it is also potentially useful in other stages of processing (identification of raw signals, groups of harmonically related signals, patterns of vehicles, etc.).

In order to perform this cooperative style of distributed processing, and thereby extend the range of applications to which distributed processing can be applied effectively, we have developed a new approach to distributed-system design. We call this new approach *functionally accurate, cooperative* (FA/C). In the following section, the FA/C/ approach is contrasted with conventional approaches to distributed-system design. Section III discusses mechanisms used

in knowledge-based artificial intelligence (AI) systems to resolve uncertainty and their appropriateness to the development of FA/C distributed systems. Section IV describes three preliminary investigations into the application of these knowledge-based AI techniques to FA/C distributed systems. Section V discusses a similar style of problem solving exhibited by management organizations and illustrates how concepts from organizational theory may be used to analyze the effectiveness of FA/C distributed systems. Section VI describes current research directions toward an improved understanding of FA/C distributed systems.

II. Functionally Accurate, Cooperative Distributed Systems

Conventional approaches to distributed-system design can be characterized by their emphasis on the maintenance of correctness in all aspects of the distributed computation. The distributed-processing system is organized so that a processing node's local data base contains appropriate portions of the overall problem-solving data base needed by the node's algorithms [5, 45]. This type of approach suggests that a distributed system be viewed as a centralized system distributed over a network, with each piece (node) in the decomposition viewed as a part of the whole system.

In these conventional distributed systems, a node rarely needs the assistance of another node in carrying out its problem-solving function. We call this type of distributed process decomposition *completely accurate, nearly autonomous* (CA/NA), because each node's algorithms operate on complete and correct information ("completely accurate") and because each node usually has in its local data base the information it requires to complete its process correctly ("nearly autonomous"). When such information is not locally available, a node requests another node to determine the required information, which is returned as a complete and correct result. In CA/NA distributed systems, this form of node interaction is often implemented using synchronous subroutine calls, in which one node is the master and the other is the slave.

The CA/NA approach, however, is not suitable for applications (such as the distributed vehicle monitoring example) in which algorithms and control structures cannot be replicated or partitioned effectively so as to match the natural distribution of data in the network. In this situation, a CA/NA system is expensive to implement because of the high communication and synchronization costs required to guarantee completeness and consistency of the local data

bases. We feel that the almost exclusive use of the CA/NA approach has restricted the types of applications that have been implemented in a distributed manner.[1]

There is an alternative and new approach to structuring distributed problem-solving systems that may be appropriate for applications in which the CA/NA approach is not suitable. In this new approach, the distributed system is structured so that each node can perform useful processing using incomplete input data while simultaneously exchanging the intermediate results of its processing with other nodes to construct cooperatively a complete solution. The hope is that the amount of communication required to exchange these results is much less than the communication of raw data and processing results that would be required using the CA/NA approach.

One way to permit a node to perform useful processing on incomplete data is to loosen the requirement that it always produce a complete and correct result. Instead, a node produces tentative results that may be incomplete, incorrect, or inconsistent with the tentative results produced by other nodes. For example, a node may produce a set of alternative partial results based on reasonable expectations of what the missing data might be. This type of node processing requires a distributed problem-solving structure that produces acceptable answers in the face of incorrect and inconsistent intermediate results. We call a system with this problem-solving structure *functionally accurate* (FA) because it exhibits acceptable system input/output behavior but is distinct from *completely accurate* problem-solving structures, in which all intermediate results shared among subtasks are required to be correct and consistent.

In an FA problem-solving structure, a node not only has to perform useful processing with incomplete input data, but also with the possibly incomplete, incorrect, and inconsistent tentative results received from other nodes. This leads to a style of problem solving in which nodes cooperate to eliminate errorful intermediate results and to converge to a complete and consistent solution. One way this can be accomplished is through an *iterative coroutine* type of node interaction, in which nodes' tentative partial results are iteratively revised and extended through interaction with other nodes. This type of node interaction suggests that such a distributed system be viewed as a *cooperative* network of interrelated tasks [39]. Therefore, we call such FA systems *functionally accurate* and *cooperative* (FA/C).

The FA/C style of processing can be characterized as problem

[1]When viewed from the perspective of the routing task alone, some algorithms used to determine message paths in a communication network work with incomplete and inconsistent views of the network [22, 60].

solving in the presence of *uncertainty*. A node may be uncertain as to what input data it is missing, the missing values of the data, and the correctness, completeness, and consistency of the results of its processing and of the processing results received from other nodes. In order to resolve these *data uncertainties* a node must be able to:

1. detect inconsistencies between its tentative partial results and those received from other nodes

2. integrate into its local data base those portions of other nodes' results that are consistent with its results

3. use the newly integrated results to make up for its missing input data so that its tentative partial results can be revised and extended.

Because consistency checking is such an important part of the FA/C approach, it is natural to think of dealing with distribution-caused uncertainty and errors as an *integral part* of the network problem-solving process. In fact, additional mechanisms required to handle hardware, communication, and processing errors may be unnecessary with the FA/C approach, since uncertainty resolving mechanisms are already a part of the distributed system's problem-solving structure [4, 17, 40].

In FA/C distributed systems, it may be difficult to determine which alternative tasks are globally the most beneficial to perform without extensive inter-node communication. This *control uncertainty* is due to differences between the natural distribution of control information among the nodes in the network and the distribution of where the control decisions are made. The existence of data uncertainty (discussed above) and uncertainty as to whether information transmitted by a node is correctly received[2] further exacerbates this difficulty.

One way to allow the distributed system to make control decisions without complete control information is to have node activity be *self-directed*. Each node uses its local estimate of the state of network problem solving to control its processing (i.e., what new information to generate) and its transmissions to other nodes [40]. The degree of self-directed activity in an FA/C system is potentially quite large because a node is able to choose a processing direction for which

[2]In some distributed communication networks, the usable capacity of the communications channel is significantly degraded if the correct reception of all messages needs to be verified. Therefore, systems that can function effectively without the acknowledgment of messages may be advantageous.

all the necessary data may not be available or consistent with other nodes. For instance, if a node does not receive an appropriate partial result in a given amount of time, it has the option to continue processing, utilizing whatever data are available at that time or to choose some other processing direction that appears to be more beneficial. This flexibility in node processing allows node interactions to be *asynchronous* and permits significant *decoupling* of node activity.

The self-directed control decisions made by each node may lead to unnecessary, redundant, or incorrect processing. The hope is that the system still produces acceptable answers (within allowable time constraints) and that the amount of additional communication resulting from incorrect local control decisions is less than the additional communication required to provide complete control information. This hope is not unreasonable, given that the additional data uncertainty caused by incorrect local control decisions may be resolvable by the same mechanisms used to resolve data uncertainties caused by incomplete local data bases. Self-directed control has the added benefit of increased system robustness in the face of communication and node failure and increased system responsiveness to unexpected events. Based on this form of node activity, it is more appropriate to view an FA/C distributed system as being *synthesized* from individual local systems operating at each node as opposed to the decomposition viewpoint described above that is normally taken of a CA/NA system.

By focusing only on CA/NA and FA/C distributed systems we do not want to suggest that completely accurate, cooperative (CA/C) and functionally accurate, nearly autonomous (FA/NA) systems do not exist. In fact, most systems should be characterized somewhere between these four extremes. Where there exists uncertainty as to the data in the system, the use of a functionally accurate (FA) over a completely accurate (CA) approach seems appropriate (due to the FA approach's tolerance of data uncertainty). Likewise, where there exists uncertainty as to what nodes should be doing and what information they should exchange, the use of a cooperative (C) over a nearly autonomous (NA) approach seems appropriate (due to the additional processing flexibility provided by the C approach).

We believe the reason most distributed system appear to be basically either CA/NA or FA/C is that data uncertainty and control uncertainty tend to go hand in hand. The presence of data uncertainty makes it difficult to determine the appropriate interaction patterns among nodes, and the presence of control uncertainty leads to increased incompleteness, inconsistency, and error in processing results. When these uncertainties are present, the use of the FA/C

approach is appropriate. Similarly, when there is little uncertainty about the completeness and consistency of data and task processing, there also tends to be little uncertainty as to the needed interactions among nodes. In this situation the more structured (and efficient) CA/NA approach is appropriate.

In the next section we show that the FA/C approach is well suited to problems that can be represented as a *search process* requiring multiple (localized) partial decisions to arrive at a solution. These decisions should not be tightly ordered, but each decision should have some consistency relationship with other decisions. The existence of a number of alternative paths to an acceptable solution and a problem representation involving multiple levels of abstraction also facilitate the FA/C approach. AI researchers have been investigating problems with similar representation characteristics. Therefore, it is not unreasonable that methodologies developed for these AI problems may be helpful in the development of FA/C distributed techniques. We now introduce this relationship.

III. Knowledge-Based AI and Functionally Accurate, Cooperative Distributed Systems

We feel that the key to the design of FA/C distributed systems is to incorporate mechanisms that can deal with uncertainty and error as an integral part of their problem-solving approach. Knowledge-based interpretation systems, such as Hearsay-II [15, 16] and MSYS [3] are examples of systems that use algorithmic structures that can resolve uncertainty and error in this way. Problem solving in these systems involves the examination of many alternative partial solutions in order to construct a complete and consistent overall solution. This style of problem solving is required because of the uncertainty (incompleteness and noise) in input data and the use of incomplete, approximate, and inconsistent knowledge in these systems.

The exploration of alternative partial solutions takes the form of a search process in which a solution is constructed through the incremental piecing together of mutually constraining or reinforcing partial solutions.[3] These partial solutions arise both from the application of diverse knowledge to the same aspects of the problem and from the application of the same knowledge to diverse aspects of the

[3]Similar uses of the *aggregation* of partial solutions arise in systems using the locus model [51], relaxation [50, 63], and the cooperating experts [2, 27, 36] paradigms.

problem. If sufficient constraints are available during this search process, incorrect partial solutions will naturally die out because it will not be possible to piece them together into more encompassing partial solutions. In this way, uncertainty is resolved as an integral part of the problem-solving process.

In many knowledge-based systems the number of possible partial solutions is large. In general, the more uncertainty that exists, the larger the number of alternatives that must be explored. If there exists a large amount of uncertainty in input data and knowledge, a significant amount of search can be required. Therefore, it is important to focus quickly on information that constrains the search space. Hence, problem solving in these systems is often asynchronous and opportunistic: there is no *a priori* order for decision making, and decisions, if they look promising, are tentatively made with incomplete information and later reevaluated in light of new information. This type of problem solving, combined with diverse and overlapping sources of knowledge, allows a solution to be derived in many different ways (i.e., different ordering sequences of incrementally constructed partial solutions and possibly different partial solutions).

Another focusing technique used in some knowledge-based systems is to structure the search space into a loose hierarchy of increasingly more abstract representations of the problem. Using this structure a high-level partial solution developed bottom up in an opportunistic way for one aspect of the problem can be used to constrain, in a top-down manner, the search for solutions to other aspects of the problem.

To illustrate these ideas, we briefly describe two knowledge-based systems, Hearsay-II [15] and MSYS [3], that exhibit this type of problem solving. While Hearsay-II and MSYS were developed for speech understanding and vision understanding, respectively, their basic structures have general applicability and have been applied to such tasks as multi-sensor interpretation [44], protein-crystallographic analysis [14], and cryptography [46].

In the Hearsay-II speech-understanding system, the understanding of spoken utterances is accomplished by combining partial solutions derived from acoustic, phonetic, syllabic, lexical, syntactic, and semantic knowledge applied to different portions of the utterance. Each area of knowledge is encapsulated in an independent module (*knowledge source*). The interaction of knowledge sources is based on an iterative data-directed form of the hypothesize-and-test paradigm. In this paradigm, an iteration involves the creation of a hypothesis, which is one possible interpretation of some part of the data, followed

by tests of the plausibility of the hypothesis. During both hypothesis creation and testing, knowledge sources use *a priori* knowledge about the problem and previously generated hypotheses to form a context for applying their knowledge. When a knowledge source creates a hypothesis from previously created hypotheses, the knowledge source extends the existing (partial) interpretation, thereby reducing the uncertainty of the overall interpretation. Processing terminates when a consistent hypothesis is generated that satisfies the requirements of a complete solution.

In the MSYS vision understanding system, each knowledge source processes a portion of the data in terms of its own limited knowledge. Each knowledge source attempts to explain what object(s) could potentially occur in a specific part of a segmented image.

> The consensus is achieved by a network of processes (representing independent knowledge sources) that communicate via shared global variables. Each process attempts to explain a fragment of the data (a region or a few regions in a segmented scene) in terms of its own limited knowledge. The confidence of an explanation is communicated to other processes attempting to explain overlapping fragments, and may cause them to reevaluate their own hypotheses. The confidence adjustment cycle continues until equilibrium is achieved [3, page 3].

When this equilibrium is achieved, a coherent set of local views has been constructed. The MSYS problem-solving technique is an example of a more general problem-solving paradigm, called *iterative refinement*, that is contained in different forms in many types of problem-solving systems [4, 50, 63, 64].

We feel that knowledge-based AI approaches to problem solving provide a basis for the development of design methodologies for FA/C distributed systems. The mechanisms used in these problem-solving systems to resolve error from incorrect and incomplete data and knowledge can also be used to structure distributed algorithms so that they work effectively with incomplete and inconsistent local data bases. We next examine these mechanisms and their implications for FA/C distributed systems (mechanism/implication).

- *Asynchronous Nature of Information Gathering/Reduced Need for Synchronization*: Problem solving is viewed as an incremental, opportunistic, and asynchronous process. In this style of problem solving, a node does not have an *a priori* order for processing information and can exploit incomplete local information.

Thus the processing order within nodes and the transmission of information among nodes do not need to be synchronized.

- *Use of Abstract Information/Reduced Inter-Node Communication Bandwidth Requirements*: The ability to use abstract information permits nodes to cooperate using messages that provide a high-level view of the system's processing without the need for detailed low-level data. This reduces the inter-node communication bandwidth needed for effective cooperation.

- *Resolution of Uncertainty Through Incremental Aggregation/Automatic Error Resolution*: Uncertainty is implicitly resolved when partial results are aggregated and compared with alternative partial solutions. This incremental method of problem solving allows a distributed system to detect and reduce the impact of incorrect decisions caused by incomplete and inconsistent local data bases and by hardware malfunction.

- *Problem Solving as a Search Process/Inter-Node Parallelism*: Because many alternative partial solutions need to be examined, parallel search by different nodes is possible. Furthermore, the additional uncertainty caused by incomplete and inconsistent local data bases can be traded off against more search. To the degree that this extra search can be performed in parallel, without proportionally more inter-node interaction, the communication bandwidth can be lowered without significant degradation in network processing time.

- *Multiple Paths to Solution/Self-Correcting Behavior*: Because there are many paths to a solution, it is possible to leave uncorrected errors that would be considered fatal in a conventional distributed system. In addition, system reliability can be improved (at the cost of additional processing and inter-node communication) without modifying the basic problem-solving structure. This variability, which is achieved through the appropriate selection and focusing of local node activity, allows consideration of additional and/or redundant paths to a solution.

Knowledge-based systems use a number of additional mechanisms to implement uncertainty resolution. These mechanisms are also important in an FA/C distributed system and include the following.

1. An integrated representation of alternative partial solutions and the coordination of partial solutions among different problem

representation levels permit the quick isolation of contradictory information.

2. Data-directed control structures allow processing to be sensitive to current relationships between alternative partial solutions and to new information.

3. Focus-of-attention strategies permit the dynamic allocation of resources among competing tasks though the evaluation of the importance of particular types of information to the problem-solving process.

4. Generator control structures (that incrementally generate credibility-ordered alternative hypotheses) reduce the possibility of combinatorial explosion during search.

5. Modular control structures (in which knowledge is structured into independent and anonymous processing modules) allow the dynamic routing of information to appropriate processing modules.

While AI paradigms provide techniques for resolving uncertainty, they have not dealt with all of the types of uncertainty that occur in a completely distributed system. Centralized global knowledge or global control has been used in these AI systems to coordinate various system modules. For example, the Hearsay-II paradigm relies on a centralized global data base (call the "blackboard") for the integration of local views generated by independent knowledge-source modules and for communication of these views to other knowledge sources. Scheduling is also centralized, based on the current hypotheses on the blackboard and a global agenda mechanism. Iterative refinement relies on either synchronization (lock-step iteration) or an explicit ordering relationship between modules in order to speed up or (in restricted cases) guarantee convergence. In addition, iterative refinement does not guarantee a consistent global solution, only a set of consistent local solutions.

It is important to reiterate, however, that even though the current formulations of these AI paradigms are not totally distributed, their ability to function with incomplete and incorrect knowledge makes them adaptable to distributed situations in which only partial and potentially inconsistent views of nonlocal information are available. The ease of this adaptability is shown in the next section in which the Hearsay-II architecture is applied to interpreting, in a distributed manner, data originating from spatially separated sensors and the iterative-refinement paradigm is applied to distributed network traffic-light control.

Other researchers have investigated different ideas for structuring unconventional distributed-processing applications. The contract-net model for distributed processing [58, 57] provides mechanisms for decentralized task allocation in an uncertain environment. Even though this model takes a CA/NA view of the sharing of results among tasks, we feel that the contract mechanism they have developed may be useful in FA/C distributed systems to provide explicit high-level decentralized coordination among the self-directed nodes to ensure greater coherence in system-wide activity.

We also agree with Sacerdoti's suggestion [53] that natural-language communication may provide a fruitful source of ideas for distributed interaction protocols that minimize communication. We are especially interested in the work on a goal-oriented model of human dialog that is based on the concepts of dialog games [42] and on model-based understanding (plan recognition) which is becoming an important part of natural language comprehension systems [7, 9, 24, 28].

There is also an emerging body of literature on decentralized control theory [35, 61] that may eventually be relevant to the development of complex FA/C distributed systems. However, current work has mainly focused on distributed control algorithms that have a CA/NA character. This emphasis has resulted in distributed control algorithms that generally require some form of high-level control to sequentialize and order the aggregation of node results. Additionally, these algorithms are restricted to decompositions in which the results of a node's decisions affect the decisions of other nodes in the network in only highly constrained ways. Due to these characteristics, these algorithms need further development to apply to FA/C distributed systems.

Based on the initial inspiration of knowledge-based AI systems, we have developed a number of prototype FA/C systems. In the next section, we discuss the design of these systems, the lessons we have learned about some of the key design issues in building FA/C systems, and the relevance of knowledge-based AI systems to these issues.

IV. Experiments in Functionally Accurate, Cooperative Distributed Processing

There have been two major thrusts to our research: to empirically evaluate the basic viability of the FA/C model of a distributed problem-solving system and to understand the strengths and weaknesses of knowledge-based AI mechanisms as a basis for FA/C

distributed systems. We have pursued these objectives by modifying a number of the uncertainty resolving techniques developed for knowledge-based AI systems for use in FA/C distributed problem-solving systems:

1. incremental hypothesize and test in the style of Hearsay-II for use in distributed interpretation systems
2. iterative refinement (successive approximation, relaxation) for use in distributed network traffic-light control
3. partially ordered hierarchical planning for use in distributed planning systems.

We have been evaluating the effectiveness of the resulting FA/C system in each of these tasks.

A. An Experiment in Distributed Interpretation

The Hearsay-II architecture appears to be a good model for an FA/C system because it incorporates mechanisms for dealing with uncertainty and error as an integral part of the problem-solving approach. Further, the processing can be partitioned or replicated naturally among network nodes because it is already decomposed into independent and self-directed modules called *knowledge sources* (KSs) that interact anonymously and are limited in the scope of the data they need and produce. It was also our hypothesis that the control and data structures of Hearsay-II could be distributed effectively because there were already existing mechanisms within its problem-solving structure for resolving uncertainty caused by incomplete or incorrect input data and KS processing.

In order to test out these hypotheses, we have been exploring the Hearsay-II architecture in distributed interpretation applications similar to the vehicle monitoring example (discussed in Section I). In these applications, each processing node can be mobile, has a set of (possibly non-uniform) sensing devices, and interacts with nearby processors through a packet-radio communication network. Nodes communicate among themselves to generate a consistent interpretation of "what is happening" in the sensed environment.

Our approach to developing a distributed interpretation architecture based on the Hearsay-II model was to organize the network into nodes operating on partial and possibly inconsistent views of the current interpretation and system state. This has led to a distributed interpretation architecture structured as a network of Hearsay-II

systems, in which each node in the network is an *architecturally complete* Hearsay-II system. "Architecturally complete" means that each node could function as a complete Hearsay-II system if it were given all of the sensory data and the required KSs. However, due to the distribution of sensory data and limited inter-node communication, each node has a limited view of the complete problem-solving data base and, in effect, a limited set of KSs. Within this basic framework, we have introduced the following additional mechanisms to support effective inter-node cooperation in a dynamic environment without high communication bandwidth.

1. To limit inter-node communication, an *incremental transmission* mechanism (with processing at each step) has been developed in which only a limited subset of a node's information is transmitted to only a limited subset of nodes. A node acts as a *generator* that transmits only a few of the most credible pieces of information and that can subsequently respond to a lack of problem-solving progress by producing alternative information.

2. To increase network reliability, a knowledge-based control mechanism, called *murmuring*, has been proposed. Here a node retransmits high-impact information if, during a specified time interval, it neither receives nor generates high-impact information. Murmuring can also be used to correct for lost communications due to intermittent channel or node failures and to bring new or moving nodes up to date.

3. To guarantee an appropriate communication connectivity among nodes, a decentralized mechanism for constructing a communication network has been developed. Using this mechanism, which relies on descriptions of the input/output (I/O) characteristics of each node, nodes act as store-and-forward message processors to provide additional connectivity. A similar mechanism can be used for the dynamic allocation of processing tasks among nodes.

4. To provide more sensitive implicit inter-node control while still retaining decentralization, each node may explicitly transmit its local control information (meta-information). Nodes can thus more directly determine the state of processing in other nodes.

Experiments were performed to determine how the problem-solving behavior of such a network of Hearsay-II systems compares to a centralized system. The aspects of behavior studied include the accuracy of the interpretation, time required, amount of inter-node

communication, and robustness in the face of communication errors. These experiments were simulations only in part, since they used an actual interpretation system analyzing real data; i.e., the Hearsay-II speech-understanding system [15].

Our goal was not to prove that one should design a distributed speech-understanding system, but rather to point out some of the issues involved in designing a distributed interpretation system dealing with incomplete and inconsistent local data. We used the Hearsay-II speech-understanding system because it has a structure that we felt was appropriate and because it is a large knowledge-based interpretation system to which we had access.

In these experiments, we modeled a spatial distribution of sensory data by having each node of the distributed speech-understanding network sample one part (time-contiguous segment) of the speech signal. The nodes in the network exchanged only high-level intermediate results. These results consisted of hypotheses (and their associated belief values) about which phrases might have occurred in the utterance under interpretation. The control decisions about which KSs to execute and what hypotheses to transmit to other nodes were made locally by each node. These local control decisions were based only on the node's local processing history and the intermediate results received from other nodes.

These network-simulation experiments have shown that the Hearsay-II speech-understanding system, with only minor changes involving the addition of some of the mechanisms described above, performs well as a cooperative distributed network even though each node has a limited view of the input data and exchanges only high-level partial results with other nodes. In an experiment with a three-node system, effective cooperation was achieved among the nodes with only 44 percent of the locally generated high-level hypotheses transmitted. This represents 77 percent of the number of high-level hypotheses created in the centralized runs. No low-level hypotheses or raw-speech data were exchanged. The three-node experiments showed an overall speedup of 60 percent over the centralized version.

In order to assess the robustness of the network system with respect to communication errors, experiments were run in which messages received by a node were randomly discarded with a specified probability. This served to model communication systems with good error detection but poor correction facilities (such as packet radio). Selection at the receiving end allowed for cases in which a broadcast message is received successfully by some nodes but not others.

In these experiments, system performance degraded gracefully

with as much as 50 percent of the messages lost. The system in many of these cases corrected for lost messages by either deriving the missing information in an alternative way or by constructing the solution in a different fashion. In summary, the system's performance with a faulty communication channel lends credence to our belief that, by making uncertainty-resolving mechanisms an integral part of network problem solving, the distributed system may be able to deal automatically with types of errors that were not anticipated during the initial design of the system. It also indicates that a trade off can be established between the amount of processing and the reliability of communication.

These experimental results support our general model of FA/C distributed-system design. They also indicate that the Hearsay-II architecture is a good one to use as a basis for this approach. A complete discussion of this work is contained in [40].

B. Distributed Network Traffic-Light Control

A second study concentrated on investigating the suitability of iterative refinement (IR) as the basis for an FA/C distributed approach to automotive traffic-light control. In our version of this task, a processor, located at each intersection, decides the setting for the traffic lights at its intersection. In order to make these decisions, each processor uses data from sensors that measure the traffic flow entering the intersection. Processors can also directly communicate with processors at neighboring intersections.

A number of test programs were developed that simulate distributed iterative-refinement algorithms for traffic control in which the knowledge applied at each node (intersection) is similar to that used in a standard centralized traffic-control system called SIGOP-II [41]. This traffic-control algorithm was chosen because it employs a serial version of the method of successive approximations and permits an easy spatial decomposition for parallel processing. Two classes of IR algorithms were studied: single-label IR (successive approximation) and multi-label IR (relaxation). In single-label IR, each node considers only one possible setting (label) for its traffic lights during each decision iteration. This is in contrast to multi-label IR where all possible traffic-light settings (labels) are considered simultaneously during each iteration. Parallel successive approximation has been explored in previous work on asynchronous iterative methods [4], but not in applications involving nonlinear, discontinuous, and non-convex cost functions. Parallel relaxation has been explored in previous work in image processing [25, 64], but not in

distributed domains that have complex compatibility relationships between nodes and significant interactions between widely separated nodes in the network. Distributed network traffic-light control has all of these characteristics.

Experiments with these simulations on arterial traffic networks show that good, but not optimal, solutions can be generated [6]. The major difficulties with the single-label IR approach have been the need for significant synchronization, environmental updating,[4] coordination techniques to prevent oscillation of the algorithms, and the inability to guarantee convergence to reasonable solutions. The major difficulties with the multi-label IR approach have been the large number of labels (i.e., the size of the search space) needed for the technique and the inability of nonlocal evidence to raise the rating of a key alternative due to *premature* reduction of the alternative's rating on the basis of nearby evidence.

In general, we have found it difficult to reproduce the performance of the sequential SIGOP-II version without significant inter-node communication and synchronization. These problems are directly attributable to changes in the centralized SIGOP-II control structure caused by the introduction of a distributed control structure. In the centralized version, a global node ordering for the refinements is precomputed using a maximal spanning tree based on the traffic volume at each node. We have found through numerous experiments that this ordering is essential in reducing the effects of non-neighboring interactions among nodes. A more detailed discussion of this research is contained in [6].

It appears that the power of the IR approach is limited because a node utilizes no global state information other than the traffic flow structures indicated by the node's immediate neighbors' traffic-light settings. Thus the amount and type of information used by a node to make decisions is severely limited. A single-label IR algorithm, for example, repeatedly makes a single decision using only information available locally or from its immediate neighbors. It does not consider alternatives or utilize a history of previous decisions. Although the multi-label IR algorithms do consider alternatives, they still do not utilize a history of previous decisions.

[4] Non-neighboring interactions among traffic-light settings are transmitted through an *environment*, the traffic flow structure, that is represented by auxiliary state variables in a formal description of the problem. This modeling permits a natural spatial decomposition of the problem involving direct interactions between neighboring signal controllers only. Unfortunately, environmental updating is often necessary because a change in control at one node often has nonlocal environmental effects that must be computed before searches by other nodes can be accurately performed.

Current research is aimed at introducing, in a distributed way, additional coordination between nodes to eliminate these problems. We are examining such techniques as multilevel relaxation [63] and distributed versions of the maximal spanning tree heuristics. In addition, we are looking into game theory [20, 48, 47] for new ideas. It is possible to view the distributed processors (signal controllers) solving the network traffic-light control problem as players of a game. The traffic-light control problem, in game-theoretic terms, is an n-person, nonzero-sum game. Already we have found that a game theory perspective of the problem leads to the use of similar coordination techniques that were utilized in our previous experiments with parallel single-label IR to control oscillation.

It is hoped that additional local processing might in some way substitute for explicit coordination between nodes. For example, in SIGOP-II the local solution at a node is a single label representing the hypothesized traffic-light setting for the node's intersection. Maintaining a history of the hypothesized labels at nodes may make it possible to eliminate some of the explicit nonlocal coordination. Similarly, introducing labels that represent not only a node's intersection setting, but also the settings of neighboring intersections (nonlocal partial solution), may also eliminate some explicit nonlocal coordination.

C. Distributed Planning

Experience with the two distributed interpretation applications has led us to understand that distributed focus of attention is a crucial aspect of all FA/C distributed systems. Distributed focus of attention involves the dynamic allocation of processing, power, memory, data, and communication resources within the distributed system. Focus of attention is a type of planning that is directed at a system's immediate internal processing. Thus, we are investigating issues in distributed focus of attention by working on the larger issue of distributed planning.

An initial investigation of distributed planning was made using Sacerdoti's NOAH planning system [52]. NOAH was selected as a suitable candidate for distribution for several reasons.

1. In NOAH, the determination of planned actions (plan development) is separate from the detection and elimination of interactions between the planned actions (plan criticism). This separation allows plan development to be performed locally, prior to the necessarily nonlocal analysis of interactions between actions planned by separate nodes.

2. Plans in NOAH are both partially ordered and hierarchical. The partial ordering of actions in NOAH eliminates the need to make action sequencing decisions until there exists a reason to make the decision. NOAH's hierarchical planning process, in which a high-level plan is developed before proceeding to increasing levels of detail, can help to generate a plan with less search than would be required if all details were considered from the outset. These techniques for reducing the combinatorial growth of planning also potentially lower the amount of inter-planner communication required in a distributed setting.

3. Because the actions planned by NOAH remain partially ordered until increased sequencing is required, the plans are well suited for parallel distributed execution without the need for additional processing to detect potential parallelism.

To complete the distribution of NOAH, its *world model* (the planner's simulation of the effects of planned actions on the environment) and plan criticism mechanisms had to be distributed. Each planner (node) is provided with a consistent initial world model that has enough detail to perform local plan development. As this world model is changed during local planning, the changes are communicated to other relevant planning nodes. When a planner receives world-model changes from other nodes, it revises its own world model and determines whether any of its locally planned actions invalidate the received world-model changes. If they do, an attempt is mode to sequence its planned actions with the planned actions of the other node (i.e., to establish a timing relationship between the actions) so that the actions no longer interfere with each other. A set of inter-planner protocols have been developed that accomplish this ordering and detect situations where a suitable ordering cannot be established.

Two important ideas were identified during this research. The first idea is the relationship between planned actions and the *resources* required to perform them. Actions interact through conflicts in resource allocation: actions may require the temporary use of particular resources (i.e., the resources are used during the action) or permanent use (the resources are required *in* the resulting world state itself). Actions may also free up resources that were previously in use. In a simple world of blocks lying on a table top, the top surface of blocks and the table top are the modeled resources. A stacking action can result in all three types of resource changes. For example, if block `A` is moved from the top of block `B` to the top of block `C`, the top of `A` must be clear throughout the action (no blocks

can be on top of A), the top of B is made clear as a result of the action, and the top of C is no longer clear as a result of the action. Resolving plan interactions is, in effect, *scheduling* resource usage in the developing plan. The identification of resource scheduling as an inherent component of the planning process suggests that distributed synchronization techniques may be applicable in the planning domain and vice versa.

The second idea is the role that *spatial locality* plays in the distributed-planning process. Since plan interactions occur via resource usage and because resources (often) exist in physical space, the spatial knowledge of local-resource requirements and their relationship with requirements of other planners can be used to reduce the amount of inter-planner communication required to detect nonlocal resource conflicts. For example, a simple scheme is to announce at periodic intervals the smallest enclosing area of all current local-resource usages. Only usages that overlap with another planner's announced area need be checked for possible plan interaction. Of course, if the announced area of another planner is enlarged, additional resource usages may have to be checked. Such a simplistic scheme breaks down, however, in situations where the areas change dramatically (such as with mobile robots) or where resources have a wide spatial area (such as a broadcast channel). The determination of a balance between the acquisition of spatial resource-usage information and the overestimation of potential resource-conflict areas is an important design issue in distributed-planning applications. A more detailed presentation of this research is contained in [10].

The three experiments in distributing the hypothesize-and-test, iterative refinement, and nonlinear hierarchical planning paradigms discussed in this section indicate that knowledge-based AI techniques are potentially useful in FA/C distributed systems. However, much research needs to be done to understand why certain algorithms tolerate the various kinds of uncertainty present in distributed problem-solving systems better than others. In the next section we discuss a first step in this direction. Using concepts from organizational theory, we describe the characteristics of algorithms that relate to the differences in their ability to handle uncertainty. In Section VI, we outline a number of research issues important in the development of a better understanding of FA/C distributed systems.

V. Organizational Theory and Functionally Accurate, Cooperative Distributed Systems

In studying management organizations, organizational theorists have worried about how *decision making under uncertainty* can be handled by various types of organizational structures. For example, Galbraith [21] has developed a set of paradigms for redesigning an organizational structure to cope with the increased communication caused by uncertainty (such as unexpected events and errorful information).

Galbraith draws upon Simon's work [55, 56] that recognized the limited information-processing capabilities of humans. Called *bounded rationality*, this limitation applies to both the amount of environmental (sensory) information that can be effectively used to make decisions and the amount of control that can be effectively exercised. Bounded rationality has severe implications on the quality of decision making when a large amount of uncertainty is present, for "the greater the task uncertainty, the greater the amount of information that must be processed ... to achieve a given level of performance" [21, page 4]. A motivation for variations in organizational structures (in terms of the type, frequency, and connectivity pattern of information flow) is to provide additional information processing capacity (to handle greater uncertainty) within the bounded rationality of the organization's individual members.

The concept of bounded rationality also applies to FA/C distributed computational structures and, in particular, can be used to analyze their ability to handle uncertainty.[5] We can characterize the rationality bounds of a node in a distributed system by looking at the scope of its local decisions (*control bounds*), the information that is used to update these decisions (*interpretation bounds*), and the updating process (*bounds on the nature of decision making*). The specific attributes of these characteristics that are important in analyzing a node's rationality bounds include the following.

- *Control Bounds*: What is the range of the environment for which a node makes a decision? What is the amount of detail (level of abstraction) of this decision? What is its accuracy? Is the decision made explicitly or implicitly through the modification of other decisions?

- *Interpretation (Sensory) Bounds*: What is the range of the environment that can be used effectively by a node for decision mak-

[5] Fox [18] has also explored the effect of bounded rationality on computational structures. His focus, however, has emphasized the structure of communication between modules rather than the specifics of the internal processing of modules.

ing? What is the detail of this environmental information? What is its accuracy? Is the information explicitly or implicitly available?

- *Bounds on the Nature of Decision Making*: How much information about the history and future goals of the decision-making process is available to a node? Is there only a single decision under consideration at a time or are alternatives considered simultaneously? What is the detail and accuracy of this information?

To illustrate these ideas, we compare distributed Hearsay-II (DHS2) and distributed iterative refinement (DIR), two algorithmic structures used in the experiments discussed in Section IV. Assuming sufficient processing power and memory to execute both algorithms in a specified amount of time, we can characterize the DHS2 algorithm as having less bounded rationality (and therefore greater potential uncertainty resolving power). This more extensive rationality can be attributed to the following differences.

Control Differences: In DIR, a node only makes a decision for its local environmental area. This decision is fully detailed and (hopefully) increasingly precise over the decision made during the previous iteration. In DHS2, a node makes decisions over varying ranges, levels of abstraction, and with varying accuracies, eventually making a decision that spans the entire environment. DHS2 therefore generates a globally coherent solution, while DIR generates locally coherent solutions.

Interpretation Differences: In DIR, only the local environment is explicitly available to a node. This information is fully detailed and as accurate as the sensor can provide. Additional environmental information *may* be implicitly available to the degree that decisions received from neighboring nodes are influenced by their local sensing (and their decisions influenced by their neighbors, and so on). DHS2 encompasses all the interpretation capabilities of DIR and, additionally, can explicitly incorporate nonlocal sensory information of varying range, abstraction, and accuracy.

Nature of Decision Making Differences: In DIR, the reasons for a particular decision are not remembered, but are only implicitly incorporated into the resultant decision. Only a single decision is under consideration by a node at a given time. This results in a *history-free* decision-making process that is Markovian in character. DHS2 provides for the explicit linkage of the decisions leading to a particular decision. Alternative competing decisions and their relationship to each other are also explicitly available.

From our experience with the network traffic-light control domain, the rationality bounds of the DIR algorithm were inadequate to handle the uncertainty caused by nonlocal interactions among the nodes' decisions. On the other hand, the DHS2 experiment was effective in an environment with even stronger nonlocal interactions (that potentially span the entire network). By keeping explicit track of the partial solutions that make up larger partial solutions, nonlocal interactions among subproblems can be correctly handled by the DHS2 algorithm.

The design strategies used by an organization to handle the increased information processing requirements of decision making (caused by uncertainty) are also relevant to FA/C/ distributed computational structures. Four design strategies are used by organizations [21].

- *Slack Resources*: An organization can reduce its need for information processing by decreasing its level of performance (using additional resources—time, equipment, personnel, etc.—or reducing the quality of performance).
- *Self-Containment*: An organization can reduce the need for information processing by choosing another decomposition in which tasks are more self contained.
- *Vertical Information Processing*: An organization can increase its capacity to process information by collecting information at the points of origin and directing it to the appropriate places in the organization and by the use of abstraction.
- *Lateral Relations*: An organization can increase its capacity to process information by placing in direct contact individuals that share a common problem.

These strategies take the following forms in the internal processing structure of a node in an FA/C distributed system.

- *Slack Resources*: By using a search process in which partial alternative decisions are incrementally made over time, a node can avoid the need for enough information to make a timely, complete, and accurate decision. The search is performed at the expense of making additional (unnecessary and redundant) tentative decisions.
- *Self-Containment*: Decision making at a node is self directed. A node attempts to do the best that it can with the information it has.

- *Vertical Information Processing*: Decision making at a node is opportunistic. The search space is structured into a loose hierarchy of increasingly more abstract representations of the problem. Using this structure, a high-level partial solution developed in an opportunistic way for one aspect of the problem can be used to constrain the search for solutions to other aspects of the problem.

- *Lateral Relations*: A node can integrate decisions asynchronously received from any other node. These decisions can be at any level of abstraction.

We feel that through a study of the literature on organizational theory and of successful organizational structures, ideas can be obtained for the design of FA/C distributed systems. This approach has already proved useful. The cooperating experts paradigm used by Lenat [36] has as its basis a protocol analysis of a group of experts solving problems, and recent work by Fox [19] has shown the similarity between organizational theories and the design of complex knowledge-based AI systems. We also believe that ideas from the areas of organizational planning and group problem solving [29, 32] may provide a source of techniques and metaphors for distributed planning.

Management organizations are only one example of natural systems that can be characterized as FA/C distributed systems. Theories describing other natural FA/C distributed systems may also be of benefit to the development of FA/C distributed-processing systems. An example of one such system is a honey-bee colony. Recent work by Reed and Lesser [49] has shown how the division of labor techniques used by the bees may provide insights into techniques for distributed focus of attention in FA/C distributed systems.

VI. Current Research Directions

Our research in the development of FA/C distributed problem-solving techniques has produced promising results. This work has highlighted many key issues that a general theory for FA/C distributed systems must address. These issues include the following.

- *Problem Decomposition*: How should the overall problem-solving task be broken into subtasks to minimize communication requirements, limit the complexity of any given subtask, and increase the reliability and performance of the overall system?

Should the decomposition be static or evolve dynamically, based upon the current status of the system?

- *Obtaining Global Information*: What nonlocal aspects of the system need to be seen by individual nodes? What levels of abstraction are appropriate for representing this information? How can potentially inconsistent and errorful nonlocal information be aggregated to form a usable nonlocal view? How should nonlocal information be held—should a single complete copy be distributed throughout the system or should each node have the portion of this information that it requires? Do multiple copies of nonlocal information need to be completely consistent, or can the system perform without complete consistency assumptions? How can dynamically changing data be represented?

- *Planning and Plan Execution*: How is planning and focus of attention performed in the system? Should planning and focusing be performed in an FA/C distributed fashion? How can the activity of nodes having overlapping information be coordinated in a decentralized and implicit way so as to control redundant computation? How can a node decide locally that it is performing unnecessary computation, selecting the aspect of the overall problem on which it should instead focus its attention?[6] How are plans executed in the system? What degree of synchronization between nodes is required during planning and plan execution in a given application?

- *Monitoring and Plan Modification*: How can the system monitor in a distributed way its success in achieving its goals? How can the system modify its current course of action in the event of an unexpected change in the environment or within the system itself? How can the system decide whether to modify its existing plan or generate a new plan based on the cost/benefit estimated for each?

- *Reliability*: What and how much uncertainty (error) can be handled using FA/C computational structures? What is the cost, in processing and communication, required to resolve various types of uncertainty? Is there the possibility that the system can

[6]This is the problem of dynamic allocation of information and processing capabilities of the network. The issue is also related to the classical allocation problem in networks: how to decide if the cost of accessing a distant data base is too high and whether, instead, the processing should be moved closer to the data or the data moved closer to the processing.

get into a severely degraded state due to the failure of a single component [43]?

- *Task Characteristics and Selection of an Appropriate Network Configuration*: What characteristics of a task can be used to select a network configuration appropriate for it? When can implicit control and information-flow structures be used? Similarly, when should flat, hierarchical, or matrix configurations, or mixtures of them, be used?[7]

We believe that answers to many of these issues will be found through the development of formal models for characterizing the uncertainty present in a task and system environment and the uncertainty-resolving power of algorithms in terms of the type and degree of uncertainty in data and control they can resolve. As a first step in this direction, we are developing a formal model for Hearsay-II-like systems [37]. We also feel that research on new forms of adaptive decentralized control (that integrate both implicit and explicit forms of control) and techniques for distributed planning and organizational self-design are vital to the development of FA/C distributed-system methodologies. The following sections outline some of our current research directions that begin to address these more generic research issues.

A. Distributed Search

Experiences with the distributed applications discussed in Section IV have led to the conjecture that all FA/C distributed problem-solving structures have at their heart distributed search. Therefore a crucial aspect of any FA/C distributed-system model is a characterization of distributed search. Distributed search involves the integration of partial results emanating from multiple semi-independent loci of search control. An adequate model for describing and analyzing distributed-search techniques has not been developed.

We feel that a model for distributed search must provide a common framework for addressing the following questions.

- *Structure of the Space of Possible Solutions*: What is the size of the space? What is the relationship (connectivity) between states in the space? What is the density and distribution of acceptable solution states in the space?

[7]Candidate characteristics include the patterns of node interaction, the type, spatial distribution, and degree of uncertainty of information, interdependencies of partial solutions, size of the search space, desired reliability, accuracy, responsiveness and throughput, and available computing resources.

- *Representation of the Search Space*: How is the search space represented in the system: as a single-level space or as a multi-level space encompassing multiple levels of abstraction? If multilevel, what is the relationship between the levels?

- *Representation of Partial Results*: How is a partial result represented in the system? What is the relationship between the representation of a partial result and the representation of the search space? What are the levels of the search space and the number of states encompassed by a partial result? What is the relationship between partial results? How are partial results extended and merged together?

- *The Search Process*: How is the search space searched?[8] What is the overlap between the local searches? What is the interaction between choices made by the local searches? How are the local searches coordinated? Are the local searches performed synchronously or asynchronously of one another? What is the nature of the communication required between the local searches (level of abstraction and scope)? What types of uncertainty can the search process resolve? What are the criteria for search termination? How optimal is the search?

We hope this model will have a taxonomic character that will provide a framework in which new alternative search techniques become apparent. The model may also lead to the development of a small set of control primitives and data structures that are appropriate for all types of distributed search techniques and applications.

B. Explicit Versus Implicit Approaches to Control

In our model of FA/C distributed systems we have emphasized an "implicit" form of decentralized control. The degree of implicit versus explicit control in inter-node coordination can be characterized by the precision with which a node can specify the nature of the tasks that are to be executed by another node and the degree to which those tasks must be performed by the other node. From a communication perspective, this control spectrum takes the form of the assumptions a node can make about who is going to receive its messages, how its messages are going to be processed, who is going to send it messages, the nature of the information contained in these

[8]Current characterizations of search using such terms as breadth-first and depth-first are inadequate even in centralized environments.

messages, what processing is expected by the transmitters of these messages, and what responses they expect to receive.

We have emphasized implicit and decentralized control for FA/C distributed systems because in this type of control a node has fewer built-in assumptions about the nature of inter-node coordination. This permits nodes to be more adaptive and flexible in the face of data and control uncertainty. We have implemented this form of control by having *self-directed* nodes that are activated in a *data-directed* manner. In this control regime, nodes interact only through the transmission of data. When a node receives information, it must decide whether or not to accept the information, what credibility to associate with it, what processing results (*goals*) it should achieve in light of this information, and what processing *tasks* it should execute to accomplish these goals. Because these decisions are made locally, node processing is entirely self-directed. In a similar self-directed manner, a node decides what and when information should be transmitted, based on the state of its local processing and its perception of the state of problem solving in the network.

This data-directed and self-directed approach to control can be contrasted with approaches where either goals or tasks are explicitly transmitted and with approaches where nodes are *externally directed*. By "externally directed" we mean that a node is required to perform some action in response to the receipt of a message. In these other approaches to control, nodes have less flexibility in their processing strategies. These alternative control regimes are illustrated in Table 1. The more precise the message (i.e., tasks are more precise than goals and goals are more precise than data) and the more externally directed a node is, the more explicit the form of control.

In our experiments with the distributed Hearsay-II architecture described in Section IV-A, we have observed that the data-directed and self-directed control regime used in this architecture can potentially lead to redundant and unnecessary processing. It appears that this form of control may not always provide sufficient global coherence among the nodes. There are two approaches for obtaining increased global coherence. The first is to provide each node with a better view of the state of problem solving in the network so that its data-directed and self-directed control decisions are more informed and consistent. This can be accomplished by having nodes exchange detailed meta-information about the state of their local problem solving and what they have learned about the states of other nodes. Another approach, that is compatible with the first, is to integrate more explicit forms of control into network problem solving. These types of control can be used to institute more nonlocal and precise

More explicit →				
	Data-directed	Goal-directed	Task-directed	More explicit →
Self-directed	• Receive data • Rate the data • Determine and rate goals based on the data • Determine and rate tasks based on the data	• Receive goals • Rate the goals • Determine and rate tasks based on the data	• Receive tasks • Rate the tasks	
Externally directed	• Receive data and ratings • Determine and rate goals based on the data • Determine and rate tasks based on the data	• Receive goals and ratings • Determine and rate tasks based on the data	• Receive tasks and ratings	

Table 1: Implicit and Explicit Forms of Control

control over the activities of individual nodes.

One example of a more explicit approach to decentralized control is the work of Smith and Davis on the Contract Net formalism [57]. In this approach, nodes coordinate their activities through contracts to accomplish specific goals. These contracts are elaborated in a top-down manner; at each stage, a node decomposes its contracts into subcontracts to be accomplished by other nodes. This process uses a bidding protocol based on a two-way transfer of information to establish the nature of the subcontracts and which node will perform a particular subcontract. This elaboration procedure continues until a node can complete its contract without assistance. From an FA/C perspective, the disadvantages of this approach are that it is difficult to quickly refocus the system to new events (because of the hierarchical nature of control) and that it does not really address the issue of coordinating the iterative coroutine exchange of partial and tentative intermediate results between nodes.

We believe that an integrated approach that incorporates the full range of implicit to explicit control may be required for effective problem solving in some FA/C distributed systems. This integrated approach can provide the flexibility to handle control and data uncertainty while still maintaining a sufficient level of global coherence to guarantee that acceptable solutions will be generated within given resource constraints. One approach to an integrated control regime is to incorporate more explicit goal-directed behavior into our implicit data-directed control. The priority given to goals received from other nodes versus local data-directed activity determines the degree of explicit versus implicit control present in the system. Our approach is to permit both types of coordination and to develop adaptive mechanisms for the system that dynamically determine an appropriate combination [12]. An important part of the development of this approach will be empirical studies to understand the appropriate balance between data-directed and goal-directed control.

C. FA/C Distributed Planning and Organizational Self-Design

Conventional planning systems generally require that plans be developed in a systematic fashion. For example, a major weakness in the current formulation of the distributed NOAH planning system (Section IV-C) is that it requires a systematic ordering of plan development and criticism. NOAH begins with a high-level representation of the plan (the goal) and expands that plan into a more detailed plan, that is analyzed for incompatible actions and possibly modified. The generation of the more detailed plan proceeds in the same order as the eventual execution of the actions (execution-time order). This expand-analyze cycle is repeated on the most detailed (last expanded) plan representation until the plan is sufficiently detailed for execution.

Each local planner considers only one possible local plan at a time. Because NOAH cannot simultaneously consider alternative partial plans, newly received planning decisions cannot be easily integrated into previously made planning decisions. Instead, when an incompatible planning decision (i.e., a conflict in resource usage) is received, a node must either ask the sending node to revise its decisions or the node must *backtrack* to a point where the received decision is no longer incompatible with its developing plan (and resume planning from that point). Backtracking may also involve canceling planning decisions that were announced to other planners. This in turn can cause those planners to cancel their decisions and so on, in a "domino" effect in which a number of planned actions

are deleted. (This effect is directly related to the problems plaguing the single-label iterative refinement algorithm used in the distributed network traffic-light control domain of section IV-B.) The requirement that a local planner make a *single* planning decision (without information about the decisions under consideration by other planners) can lead to much wasted planning effort and a corresponding cost in wasted communication. This style of processing does not foster a form of cooperation that can effectively deal with incomplete and inconsistent local planning data bases.

The Hayes-Roth cognitive model of planning [26] is an example of a style of planning in which a number of alternative, competing, partial plans are developed concurrently. Based on a Hearsay-II architecture [15], their planning model is hierarchical but not strictly limited to a top-down execution-time-ordered planning sequence. Instead, planning proceeds *opportunistically*, with each new planning decision integrated into a subset of previously made decisions. New decisions may also produce independent, competing, partial plans at various levels of abstraction. As planning continues, some of these partial plans die out and others are merged together into larger, more complete plans. Planning terminates when an acceptable overall plan is developed.

In an environment where unpredictable external information is asynchronously received, an integrated representation of competing and cooperating partial plans allows the planner to retain relevant portions of previous planning results. This style of incremental plan modification allows better refocusing in light of new information than does backtracking. It is also important in a distributed environment to attempt to quickly reduce the size of the planning (search) space, since additional search requires additional communication. The opportunistic behavior of the Hayes-Roth model allows greater flexibility in solving crucial aspects of the planning task before proceeding to less crucial aspects.

The success of distributing the Hearsay-II speech-understanding system [40] suggests that the Hayes-Roth planning model can serve as a basis for the development of an FA/C distributed planning organization. However, a number of issues relating to the use of partial and inconsistent plans in such a framework remain to be resolved.

Planner interactions can also be reduced by the selection of action sequences that are applicable in a range of situations over alternative sequences that are less stable but perhaps more execution-time efficient. Most planners concentrate on finding optimal execution-cost or *solution-optimal* plans (after Sproull [59, page 60]). Sproull's

notion of *planning-optimal* plans, in which the cost of the planning *and* execution of the plan is optimized, is particularly apropos in the distributed-planning environment. An FA/C style of planner would attempt to identify stable action sequences based on its expectations of the actions that other planners might perform and on anticipated environmental changes. Understanding how portions of plans interact during planning and plan execution can provide measures of plan stability that can be used to evaluate alternative courses of action based on these expectations.

Given that the cost of communication is relatively expensive in comparison to the cost of processing in a distributed system, it is important to regard all aspects of the environmental spectrum as potential communications media. The typical distributed-system viewpoint assumes that the message-transfer channel is the sole conduit of information exchange in the system. However, observation of another planning system's executing plan can provide a variety of implicit information. Consider a room with a tall ladder and two repairmen with the task of replacing a ceiling light. If one repairman grabs the ladder to steady it, the other would probably assume it was his responsibility to climb up and replace the light—without any need for verbal communication. The application of planning (common sense reasoning) to the interpretation of another's actions is currently receiving attention from language researchers in the analysis of natural language utterances [7, 9, 24, 28]. However, the interleaving of plan development and plan execution in a distributed environment leads to a number of unresolved technical questions.

An assumption made in the distributed NOAH planning system was the existence of a mechanism for allocating planning activity to individual nodes. Different allocations of planning activity (even for the same planning problem) produce significant differences in the complexity of the planning process. In fact, the allocation of planning activity is part of the larger issue of determining an appropriate organizational structure for the particular distributed problem-solving situation.

The *organizational structure* of a distributed system is the pattern of information and control relationships that exist between the nodes in the system and the distribution of problem-solving capabilities among the nodes in the system. Organizational structures include hierarchies, heterarchies or flat structures, matrix organizations, groups or teams, and market or price systems. *Organizational design* is the explicit planning of these inter-node relationships.

The organizational structure of a distributed system relates strongly to its effectiveness in a given problem-solving situation.

This effectiveness is a multivalued measure incorporating such parameters as processing resources, communication requirements, timeliness of activity, accuracy of activity, etc. An organizational structure may lose effectiveness as the internal or external environment of the distributed system or the nature of the problem-solving task changes. In order to respond to such a change, the distributed system must detect the decreased effectiveness of its organizational structure, determine plausible alternative structures, evaluate the cost of continuing with its current structure versus the cost of reorganizing itself into a more appropriate structure, and carry out such reorganization if appropriate.

Organizational-design decisions are faced regularly in human organizations, especially those in the business community where pressures of efficiency are most severe. Theories of organizational design which attempt to explain the art of organizational structuring in these human organizations are highly relevant to the development of organizational-design knowledge for distributed systems.

Research outlined in [11] bypasses further refinement of distributed-planning techniques, such as those developed in distributing NOAH, in favor of the development of a framework that encompasses both planning and organizational self-design. Although many issues in distributed planning remain to be solved, the development of a technology for distributed organizational self-design seems the more salient research direction. A rephrasing of the programming adage "Don't optimize a bad algorithm—rewrite it" seems appropriate: "Don't work to improve plans within a bad organizational structure—*reorganize*."

VII. Conclusion

We feel that methodologies can be developed for functionally accurate, cooperative (FA/C) distributed systems in which the distributed algorithms and control structures function with both inconsistent and incomplete data. These methodologies are necessary in order to extend the range of applications that can be effectively implemented in distributed environments.

FA/C problem-solving structures are also important in the implementation of complex applications in centralized environments. These applications are often organized in the form of a collection of independent modules. In such a structure, it can be conceptually difficult to develop, and expensive to maintain, a complete and consistent centralized problem-solving data base with which the modules interact. Techniques that permit the relaxation of completeness and

consistency requirements would be a significant aid in the development and maintenance of these (logically distributed) systems.

There are two concepts that form the basis of FA/C distributed methodologies:

1. to view an FA/C distributed system as a network of cooperating systems that share common goals, where each system is able to perform significant local processing using incomplete and inconsistent data
2. to handle the uncertainty in control, data, and algorithms introduced by distribution as an integral part of the network problem-solving process.

Techniques developed in the context of knowledge-based AI systems and organizational theory provide a basis for implementing both concepts.

Acknowledgment

The authors would like to acknowledge the research efforts of Lee Erman on distributed Hearsay-II and Richard Brooks on network traffic-light control that are discussed in this article. We would also like to acknowledge discussions with the other members of the University of Massachusetts Cooperative Distributed Problem-Solving Research Group: Jasmina Pavlin, Scott Reed, and Ram Mukunda. Michael Arbib, Randy Davis, Lee Erman, and Reid Smith also provided helpful comments on early drafts of this article.

References

[1] Michel E. Adiba, Jean-Claude Chupin, Robert Demolombe, Georges Gardarin, and Jean Le Bihan. Issues in distributed data base management systems: A technical overview. In S. Bing Yao, editor, *Proceedings of the Fourth International Conference on Very Large Data Bases*, pages 89–110, West Berlin, Germany, September 1978. IEEE Computer Society.

[2] Michael A. Arbib. Artificial intelligence and brain theory: Unities and diversities. *Annals of Biomedical Engineering*, 3:238–274, 1975.

[3] Harry G. Barrow and Jay M. Tenenbaum. MSYS: A system for reasoning about scenes. Technical Report 121, SRI International, Menlo Park, California, April 1976.

[4] Gerard M. Baudet. Asynchronous iterative methods for multiprocessors. *Journal of the ACM*, 25(2):226–244, April 1978.

[5] P. A. Bernstein, J. B. Rothnie, Jr., N. Goodman, and C. A. Papadimitriou. The concurrency control mechanism of SDD-1: A system for distributed databases (the fully redundant case). *IEEE Transaction on Software Engineering*, SE-4(3):154–168, May 1978.

[6] Richard S. Brooks and Victor R. Lesser. Distributed problem solving using iterative refinement. Technical Report 79-14, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts 01003, May 1979.

[7] Bertram Bruce and Dennis Newman. Interacting plans. *Cognitive Science*, pages 195–233, 1978. (Also published in *Readings in Distributed Artificial Intelligence*, Alan H. Bond and Les Gasser, editors, pages 248–267, Morgan Kaufmann, 1988.).

[8] W. W. Chu. *Advances in Computer Communications*. Artech House, Dedham, Massachusetts, 1977.

[9] P. R. Cohen and C. R. Perrault. Elements of a plan-based theory of speech acts. *Cognitive Science*, 3(3):177–212, July 1979.

[10] Daniel D. Corkill. Hierarchical planning in a distributed environment. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pages 168–175, Tokyo, Japan, August 1979. (An extended version was published as Technical Report 79-13, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts 01003, February 1979.).

[11] Daniel D. Corkill. An organizational approach to planning in distributed problem-solving systems. Technical Report 80-13, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts 01003, May 1980.

[12] Daniel D. Corkill and Victor R. Lesser. A goal-directed Hearsay-II architecture: Unifying data-directed and goal-directed control. Technical Report 81-15, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts 01003, (in preparation).

[13] D. Gerd Dimmler, *et al.* Brookhaven reactor experiment control facility: A distributed function computer network. *IEEE Transactions on Nuclear Science*, NS-23:398–405, February 1976.

[14] Robert S. Engelmore and H. Nii. A knowledge-based system for the interpretation of protein X-ray crystallographic data. Technical Report CS-77-589, Computer Science Department, Stanford University, Stanford, California 94305, February 1977.

[15] Lee D. Erman, Frederick Hayes-Roth, Victor R. Lesser, and D. Raj Reddy. The Hearsay-II speech-understanding system: Integrating knowledge to resolve uncertainty. *Computing Surveys*, 12(2):213–253, June 1980.

[16] Lee D. Erman and Victor R. Lesser. A multi-level organization for problem solving using many diverse cooperating sources of knowledge. In *Proceedings of the Fourth International Joint Conference on Artificial Intelligence*, pages 483–490, Tiblisi, Georgia, USSR, August 1975.

[17] Richard D. Fennell and Victor R. Lesser. Parallelism in Artificial Intelligence problem solving: A case study of Hearsay II. *IEEE Transactions on Computers*, C-26(2):98–111, February 1977. (Also published in *Readings in Distributed Artificial Intelligence*, Alan H. Bond and Les Gasser, editors, pages 106-119, Morgan Kaufmann, 1988.).

[18] Mark S. Fox. Organization structuring: Designing large complex software. Technical Report 79-155, Computer Science Department, Carnegie-Mellon University, Pittsburgh, Pennsylvania, December 1979.

[19] Mark S. Fox. An organizational view of distributed systems. In *Proceedings of the International Conference on Cybernetics and Society*, number 354-359, Denver, Colorado, October 1979.

[20] Avner Friedman. *Differential Games*. Wiley, New York, 1971.

[21] Jay Galbraith. *Designing Complex Organizations*. Addison-Wesley, 1973.

[22] M. Gerla. Deterministic and adaptive routing policies in packet-switched computer networks. In *Proceedings of the Third ACM Data Communication Symposium*, pages 23–28, November 1973.

[23] Peter E. Green. Private communication. Technical report, M.I.T. Lincoln Labs, Cambridge, Massachusetts, 1979.

[24] Barbara Grosz. Utterance and objective: Issues in natural language communication. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pages 1067–1076, Tokyo, Japan, August 1979.

[25] Allen R. Hanson and Edward M. Riseman. Segmentation of natural scenes. In Allen R. Hanson and Edward M. Riseman, editors, *Computer Vision Systems*, pages 129–163. Academic Press, 1978.

[26] Barbara Hayes-Roth and Frederick Hayes-Roth. A cognitive model of planning. *Cognitive Science*, 3(4):275–310, October–December 1979.

[27] Carl Hewitt. Viewing control structures as patterns of passing messages. *Artificial Intelligence*, 8(3):323–364, Fall 1977.

[28] J. R. Hobbs. Conversation as planned behavior. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pages 390–396, August 1979.

[29] L. R. Hoffman. Group problem solving. In L. Berkowitz, editor, *Advances in Experimental Social Psychology*, volume 2, pages 99–132. Academic Press, 1965.

[30] T. G. Hoffman. Upgrading gauge measurement with distributed computers. *Control Engineering*, 22:39–41, 1975.

[31] R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kunzelman. Advances in packet radio technology. In *Proceedings of the IEEE*, pages 1468–1496, November 1978.

[32] H.H. Kelley and J. W. Thibaut. Experimental studies of group problem solving and process. In G. Lindzey, editor, *Handbook of Social Psychology*, pages 735–785. Addison-Wesley, 1954.

[33] S. R. Kimbleton and G. M. Schneider. Computer communication networks: Approaches, objectives, and performance considerations. *Computing Surveys*, 7(3):129–173, September 1975.

[34] R. Lacoss and R. Walton. Strawman design of a DSN to detect and track low flying aircraft. In *Proceedings of the Distributed Sensor Nets Workshop*, pages 41–52, 1978. Published by the Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pennsylvania.

[35] R. E. Larson, editor. *Tutorial: Distributed Control*. IEEE Computer Society, New York, 1979.

[36] Douglas B. Lenat. Beings: Knowledge as interacting experts. In *Proceedings of the Fourth International Joint Conference on Artificial Intelligence*, pages 126–133, Tiblisi, Georgia, USSR, August

1975. (Also published in *Readings in Distributed Artificial Intelligence*, Alan H. Bond and Les Gasser, editors, pages 161–168, Morgan Kaufmann, 1988.).

[37] V. R. Lesser, S. Reed, and J. Pavlin. Quantifying and simulating the behavior of knowledge-based interpretation systems. In *Proceedings of the First Annual National Conference on Artificial Intelligence*, pages 111–115, Stanford, California, August 1980.

[38] Victor Lesser, Daniel Corkill, Jasmina Pavlin, Larry Lefkowitz, Eva Hudlická, Richard Brooks, and Scott Reed. A high-level simulation testbed for cooperative distributed problem solving. Technical Report 81-16, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts 01003, June 1981. (Revised and shortened versions of this report appeared in *Proceedings of the Distributed Sensor Networks Workshop*, MIT Lincoln Laboratory, Lexington, Massachusetts, pages 247–270, January 1982, and in *Proceedings of the Third International Conference on Distributed Computer Systems*, pages 341–349, October 1982.).

[39] Victor R. Lesser and Daniel D. Corkill. Cooperative distributed problem solving: A new approach for structuring distributed systems. Technical Report 78-7, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts 01003, May 1978.

[40] Victor R. Lesser and Lee D. Erman. Distributed interpretation: A model and experiment. *IEEE Transactions on Computers*, C-29(12):1144–1163, December 1980. (Also published in *Readings in Distributed Artificial Intelligence*, Alan H. Bond and Les Gasser, editors, pages 120–139, Morgan Kaufmann, 1988.).

[41] E. B. Lieberman and J. L. Woo. SIGOP II: A new computer program for calculating optimal signal timing patterns. Number No. 596, 1977.

[42] William C. Mann. Toward a speech act theory for natural language processing. Technical Report RR-79-75, USC/Information Sciences Institute, Marina del Rey, California 90291, March 1980.

[43] Phillip M. Merlin. A methodology for the design and implementation of communication protocols. Technical report, IBM Thomas J. Watson Research Center, Yorktown Heights, New York, June 1975.

[44] Penny Nii and Edward A. Feigenbaum. Rule-based understanding of signals. In D. A. Waterman and Frederick Hayes-Roth, editors, *Pattern-Directed Inference Systems*, pages 483–501. Academic Press, 1978.

[45] R. W. Peebles and E. Manning. System architecture for distributed data bases. *IEEE Computer* Special Issue on Distributed Processing, 11(1):40–47, January 1978.

[46] Shmuel Peleg and Azriel Rosenfeld. Breaking substitution ciphers using a relaxation algorithm. *Communications of the ACM*, 22(11):598–605, November 1979.

[47] Anatol Rapoport. *N-Person Game Theory: Concepts and Applications*. University of Michigan Press, Ann Arbor, Michigan, 1966.

[48] Anatol Rapoport. *Two-Person Game Theory: The essential ideas*. University of Michigan Press, Ann Arbor, Michigan, 1966.

[49] Scott Reed and Victor R. Lesser. Division of labor in honey bees and distributed focus of attention. Technical Report 80-17, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts 01003, September 1980.

[50] Azriel Rosenfeld, Robert A. Hummel, and Steven W. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(6):420–433, June 1976.

[51] Steven M. Rubin and Raj Reddy. The LOCUS model of search and its use in image interpretation. pages 590–595, Cambridge, Massachusetts, August 1977.

[52] Earl D. Sacerdoti. *A Structure for Plans and Behavior*. Elsevier, 1977.

[53] Earl D. Sacerdoti. What language understanding research suggests about Distributed Artificial Intelligence. In *Proceedings of the Distributed Sensor Networks Workshop*, pages 8–11, 1978. Published by the Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pennsylvania.

[54] J. D. Schoeffl and C. W. Rose. Distributed computer intelligence for data acquisition and control. *IEEE Transactions on Nuclear Science*, NS-23:32–54, February 1976.

[55] Herbert A. Simon. *Models of Man*. John Wiley & Sons, 1957.

[56] Herbert A. Simon. *The Sciences of the Artificial*. MIT Press, 1969.

[57] Reid G. Smith and Randall Davis. Cooperation in distributed problem solving. In *Proceedings of the International Conference on Cybernetics and Society*, pages 366–371, Denver, Colorado, October 1979.

[58] Reid Garfield Smith. *A Framework for Problem Solving in a Distributed Processing Environment*. PhD thesis, Stanford University, December 1978. Also published as Technical Report STAN-CS-78-700. A revised version was published by UMI Research Press.

[59] Robert F. Sproull. *Strategy Construction using a Synthesis of Heuristic and Decision-theoretic Methods*. PhD thesis, Stanford University, Stanford, California 94305, 1977. (Also published as Technical Report CSL-77-2, Xerox Palo Alto Research Center, Palo Alto, California, July, 1977.).

[60] William D. Tajibnapis. A correctness proof of a topology information maintenance protocol for a distributed computer network. *Communications of the ACM*, 20(7):477–485, July 1977.

[61] Robert R. Tenny. *Distributed Decision Making using a Distributed Model*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, June 1979. (Also published as Technical Report LIDS-TD-938, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, Massachusetts, 02139, June 1979.).

[62] Various authors. Special issue on microelectronics. *Scientific American*, 237(3):62–245, September 1977.

[63] Steven W. Zucker. Vertical and horizontal processes in low level vision. In Allen R. Hanson and Edward M. Riseman, editors, *Computer Vision Systems*, pages 187–195. Academic Press, 1978.

[64] Steven W. Zucker, Robert A. Hummel, and Azriel Rosenfeld. An application of relaxation labeling to line and curve enhancement. *IEEE Transactions on Computers*, C-26(4):394–403, April 1977.