INCREASING COHERENCE IN A DISTRIBUTED PROBLEM SOLVING NETWORK

Edmund H. Durfee, Victor R. Lesser, and Daniel D. Corkill

Department of Computer and Information Science
University of Massachusetts
Amherst, Massachusetts 01003

ABSTRACT

Globally coherent behavior is the holy grail of distributed problem solving network research. Obtaining coherent network activity without sacrificing node autonomy and network flexibility places severe demands on the local control component of each node. We introduce new mechanisms that allow a node to compute an abstracted, high-level description of its local state which it then uses to formulate multi-step plans. Not only do these mechanisms significantly improve local problem solving performance, but they also enable nodes to make dynamic refinements to their long-term network organisation knowledge. The coordination decisions made by nodes are thus increasingly responsive to changes in network activity as problem solving progresses. We provide experimental results indicating that these new mechanisms improve the internal control decisions of a node, reduce the communication requirements of the network, and improve network coherence. We believe that these mechanisms would also be useful for control in centralised multi-level blackboard-based problem solving systems.

I. INTRODUCTION

Achieving global coherence in cooperative distributed problem solving networks (DPSNs) is a major problem [4,13]. In a DPSN, each node is an intelligent semi-autonomous problem solving agent that determines its own behavior based on its perception of network activities. Global coherence means that the activities of the nodes should appear to make sense given overall network goals. Nodes should avoid unnecessarily duplicating the work of others, sitting idle while others are swamped with work, or transmitting information that will not improve overall network performance. Because network coordination must be decentralised to improve reliability and responsiveness, the amount of global coherence in the network is dependent on the degree to which each node makes coherent decisions based on its local view of network problem solving activities.

This research was sponsored, in part, by the National Science Foundation under Grant MCS-S300230 and by the Defense Advanced Research Projects Agency (DOD), monitored by the Office of Naval Research under Contract NR049-041.

At any given time, a node will rank its pending tasks based on how it believes each will improve network problem solving. A decision by the node to execute the top ranked task is therefore more or less coherent depending on how highly ranked the task would have been if the node had a completely global view of network problem solving. Full global coherence requires that each node have a complete and accurate view of the past, present, and intended future activities of all other nodes. If this is done by globally predefining a coordinated multi-agent plan at network creation, the network will be inflexible to changing problem solving situations and network characteristics. Alternatively, having nodes broadcast all state changes and future intentions is infeasible due to bandwidth limitations and channel delays. Therefore, we have no practical means to insure full global coherence. The functionally accurate, cooperative approach to distributed problem solving develops a framework in which network goals can be achieved with only partial global coherence [13]. However, since partial coherence wastes resources and degrades performance, we have been developing mechanisms which increase coherence without significant additional communication costs.

Our previous work toward this end developed a decentralised approach to network coordination in which each node is guided by a high-level strategic plan for cooperation among the nodes in the network [3]. This strategic plan, represented as a network organisational structure, specifies in a general way the communication and control relationships among the nodes. The organisational structure increases the likelihood that nodes will be coherent in their behavior by predefining a limited range of options available to a node. Network flexibility is maintained by not limiting these options too tightly. Sophisticated local control plays a key part in this approach because decisions about which of these options to pursue must be based on short-term information about the current situation.

In this paper, we describe new mechanisms that allow a node to refine its perception of the role it currently plays in the organization. This refined view is achieved by providing each node with the ability to reason about its current state of problem solving and to make predictions about its future actions. To accomplish this, these new

mechanisms allow a node to compute an abstracted, high-level description of its local state. The node uses this description to formulate high-level goals and to generate plans to achieve them. Since each plan incorporates a sequence of actions, the pursuit of a specific plan allows the node to make reliable predictions about its actions in the near future. These predictions enable the node to make medium-term, dynamic refinements to how it views its role in the network organization, and it may modify its local processing accordingly. Furthermore, if nodes occasionally exchange *meta-level* information about these refined views of the organizational roles they will be playing in the near future, each node will have a more global view of the network problem solving activity, and global coherence will increase.

We have implemented and empirically evaluated our ideas using the Distributed Vehicle Monitoring Test bed. The next section outlines the relevant aspects of the test bed, describes how our mechanisms were incorporated, and discusses experimental results indicating improvement in local problem solving ability. We then study how these mechanisms can be used to improve network coherence, and present results indicating their utility. Finally, we relate our work to other research in distributed problem solving and discuss the implications of the preliminary research we have outlined, along with the directions we will be pursuing in the future.

n. INCREASING THE SOPHISTICATION OF A PROBLEM SOLVING NODE

The distributed vehicle monitoring test bed (DVMT) is a flexible and fully-instrumented research tool for the evaluation of distributed network designs and coordination policies [14]. The DVMT simulates a network of problem solving nodes attempting to identify, locate and track patterns of vehicles moving through a two-dimensional space using signals detected by acoustic sensors. Each problem solving node is an architecturally-complete Hearsay-n system with knowledge sources and levels of abstraction appropriate for this task. The basic Hearsay-II architecture has been extended to include more sophisticated local control [2], knowledge sources (KSs) for communicating hypotheses and goals among nodes, and data structures called *interest areas* that specify the organisational role of a node [3]. These interest area specifications are used by the local node control in deciding what problem-solving and communication knowledge sources should be instantiated and how these knowledge source instantiations (KSIs) should be rated for possible execution.

In this section, we introduce further modifications to this architecture by providing a node with the capability to generate and reason with a more complete view of its past, present, and future activities. Although nodes generally tend to methodically perform sequences of related actions, they are unable to represent and reason about such sequences. For example, given a highly rated hypothesis, a node typically executes a sequence of KSIs that drive up low level data to extend the hypothesis. However, the entire sequence of KSIs is never on the queue at once. We have therefore developed a structure, called a *plan,* to explicitly represent a KSI sequence.

A. Blackboards, Plans, and Node Activities

Each *plan* represents a desire to achieve a high-level goal by performing a sequence of activities. To identify plans, the node needs to recognize these high-level goals. Inferring high-level goals based on pending KSIs is an inappropriate strategy; it is like attempting to guess a chess opponents strategy after seeing a single move. Furthermore, the hypothesis and goal blackboards provide information at too detailed a level to infer these high-level goals. What is required is a structure similar to the blackboards that groups related hypotheses and goals into a single structure. We have developed a preliminary version of this structure which we call the *abstracted blackboard,* a multi-level structure reminiscent of the focus-of-control database first used in the Hearsay-II speech understanding system [7]. Our implementation of the abstracted blackboard is incomplete because it does not adequately incorporate the information from the goal blackboard. However, for the type of processing done in the DVMT, hypothesis abstraction is usually effective.

Hypotheses with similar level, time, and region characteristics are grouped together on the abstracted blackboard. This grouping acts as a smoothing operator, obscuring details about individual hypothesis interactions so that broader, long-term interactions between areas of the solution space can be discerned. By transforming the data blackboard into the abstracted blackboard, we explicitly generate a state representation that is uniquely appropriate for planning at a higher level of abstraction. We believe that the significant success of our modified architecture can largely be attributed to having such a representation, and expect that the control components in other multi-level blackboard-based problem solving systems might similarly find such representations useful [0j.

In our preliminary implementation, the abstracted blackboard takes the form of a two-dimensional array, with level and time indices. Each hypothesis has associated with it a sequence of time-locations which indicates where the hypothesized vehicle was at various times. When a hypothesis is created, it is incorporated into the abstracted blackboard by stepping through this sequence of time-locations and modifying the appropriate level-time entries in the abstracted blackboard. Each level-time entry contains some number of regions, and if the location associated with the specific time can be included in one of these regions (perhaps by enlarging the region within certain bounds), the hypothesis is associated with that region. Otherwise, a new region is formed for the hypothesis.

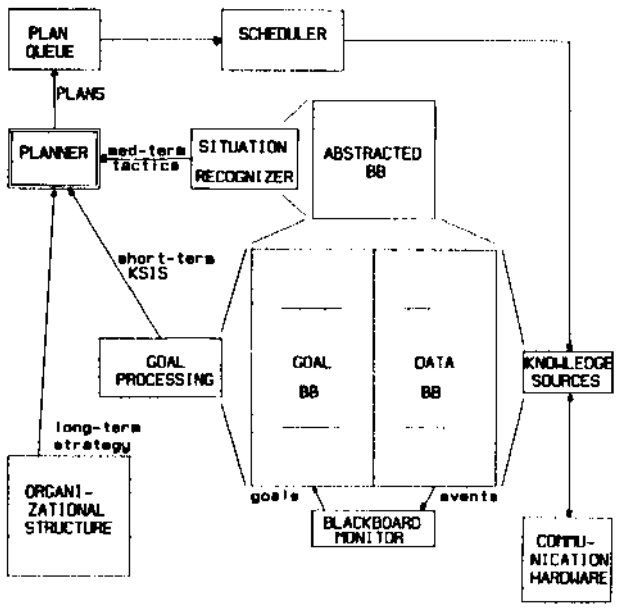
Each level-time-region of the abstracted blackboard is summarized into a set of values that are derived from the associated hypotheses. These values include the maximum belief of the hypotheses in the level-time-region,

the number of highly believed hypotheses, the number of KSIs stimulated by these hypotheses that have yet to be invoked, the total number of hypotheses in the level-time-region and how many uninvoked KSIs are associated with them, and an indication as to the other level-time-regions that share at least one of the hypotheses. This information allows the *situation recognizer* to develop a higher level view of the problem solving. For example, low maximum belief indicates the problem solving approach in that area should be re-evaluated, a large number of equally rated hypotheses could imply that there is uncertainty that should be resolved, and a large number of pending KSIs indicates the need for making an informed and judicious choice as to which action to take next. Based on this higher-level view, we can begin to form higher level goals. A goal might be to merge hypotheses in adjacent time-regions, to improve the belief of an established hypothesis, or to extend a highly believed hypothesis into a new region.

The detection of these goals, and the subsequent generation and ranking of their respective plans, is in itself a complex problem solving task. Our current implementation is a first step toward this end, in which we only consider very simple but important plans. Given the abstracted blackboard, our planner scans down it, looking for regions of high belief. Having found such a *stimulus region,* the planner determines whether there is any indication that the data in this region can be improved (this is done by determining whether any corresponding lower level regions have higher belief than the upper level regions), and if so indicated, a plan is formed to achieve this improvement. Otherwise, a plan is generated to extend this highly rated region, either by merging a hypothesis in this region with a hypothesis in an adjacent region on the same level (if any), or by driving lower level data in an adjacent area up to a level at which it can be incorporated. If none of these plans can be formed, then a plan to synthesize the hypotheses in this highly rated region up to a higher level of abstraction may be formed.

Plans in our current implementation are not yet fully developed, because a plan should not only involve the specification of an eventual goal, but also of a sequence of actions needed to achieve this goal. Only the next potential step(s) for achieving the plan are currently represented as a priority rated queue of KSIs. In turn, the node maintains a queue of plans, ordered based on their respective ratings. A plan rating is based on a number of factors, including the belief of its stimulus region, the level of its stimulus region, the interest in that region (specified by the interest areas), the ratings of its KSIs, and whether the stimulus region represents hypotheses generated locally or it represents received hypotheses (to reason more fully about potentially distracting information received from outside). Therefore, in choosing its next activity, a node will invoke the highest rated KSI in the highest rated plan.

We have therefore made important modifications to the control structure of a node (Figure 1). As the figure indicates, the creation and ranking of plans requires the planner to integrate the influences of the long-term strategy



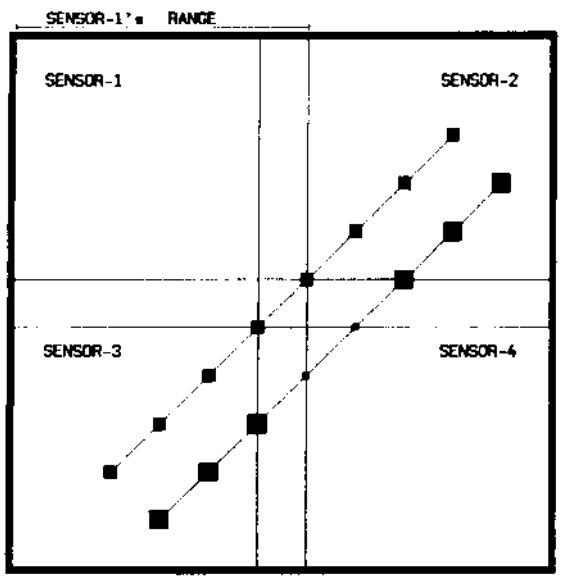
The principal problem solving components of a node are presented. Note that the planner must integrate long-term, medium-term, and short-term information.

Figure 1: The Problem Solving Architecture of a Node.

of the organizational structure (the interest areas), the medium-term higher-level view of the current situation (the abstracted blackboard), and the short-term KSI input indicating actions that can be achieved immediately (the KSI queue). Hence, decisions in a plan-based node are more informed than those in a KSI-based node (a node without plans). Moreover, a plan-based node is no less opportunistic, because plans, unlike KSIs, are interruptable. If an area outside the current plan looks more promising, a plan to work there may temporarily supplant the current plan at the top of the queue. In addition, when plans are introduced, one can begin reasoning about the time invested in a particular area, and whether it is really in the node's best interests to leave this area for another.

B. Experiments with Plan-based Nodes

We now briefly illustrate how problem solving is improved in a plan-based node (we shall consider multi-node networks in the next section). Consider the sensor configuration and input data shown in Figure 2. The vehicle track has two strongly sensed areas divided by a weakly sensed area, while the ghost track is moderately sensed throughout. In the centralised case, a single node receives data from all four sensors. If the node makes only correct decisions, it can generate the solution in 40 time units. However, the presence of moderate ghost data serves to distract the node. This distraction is severe if the node is KSI-based—the solution is found in 213 time units.



The four sensors have a small amount of overlap. There are eight data points on both the ghost track (upper) and the vehicle track (lower). The size of the data point indicates how strongly it is sensed.

Figure 2: Four Sensor Configuration.

Although a plan-based node is also distracted, its high-level view helps it quickly recognize that the distracting data will not satisfy the high-level goal which drives the plan. For example, the high-level goal may be to create a hypothesis which extends a hypothesis with three time-locations to four time-locations. Distracting data which initially looks like it might satisfy this goal is developed until the planner, using the high-level view, recognises that the high-level goal will not be achieved. At this point, a plan to develop other data which could satisfy the high-level goal becomes the highest rated plan. Due to the high-level view and the sophistication of the planner, the time spent deviating from the correct solution path is reduced, and the plan-based node can generate the solution in 58 time units.

Plan-based nodes have been used in a number of other environments with similar results. In both the centralised case and the multi-node case where nodes do not exchange met a-level information (see next section), the increased self-awareness afforded by the new mechanisms significantly improves problem solving performance. The experiments thus serve to emphasise the importance of sophisticated local control which recognises and reacts appropriately to various problem solving situations. We anticipate expanding the repetoire of situations which can be dealt with so that plans can be developed in more complex environments.

III. INCREASING THE COHERENCE OF THE PROBLEM SOLVING NETWORK

We have seen how the performance of a centralized node can be improved by allowing it to reason more fully as to the appropriate activities to perform. We now examine the multi-node case, where each node has a limited local view of network problem solving. In the previous section, we established that the problem solving behavior of a plan-based node is more effective than that of a KSI-based node. Therefore, we can expect that a network of such nodes might have improved performance, not because they display better "teamwork* (their global knowledge does not increase), but rather because each is a better "player".

These expectations were empirically verified on a number of environments. Environment El uses the configuration of Figure 2, but assigns a separate node to each sensor, each node being allowed to communicate with its neighbors. A second environment, E2, is a four-node environment identical to El except that the positions of the vehicle and ghost tracks are reversed. Note that, in this case, the weakly sensed vehicle data is received by all four nodes, and is the only data received by one of them. A larger sensor and data configuration consisting of ten sensors and eighteen sensed times was developed, patterned after the four sensor configuration [5]. A third environment, E3, consisting of ten nodes was built upon this configuration.

The experimental results are given in Table 1. Note that, in all cases, the multi-node network composed of plan-based nodes is significantly better. Environment E2 approaches optimal results because one node receives only the weakly sensed vehicle data, and so, will drive this data up earlier. In the other environments, however, work on this weakly sensed data is not as timely because the nodes prefer to work redundantly on the more highly sensed vehicle and ghost data in their overlapping sensed areas. This redundancy wastes computation time that could be used to develop the weak vehicle data instead—the network is not behaving coherently.

By transmitting the abstracted blackboards (or portions thereof), nodes can reason about the *past* activities of their neighbors. Furthermore, if a node knows the current plan of its neighbor, it can reason about the *present* actions of its neighbor. Reasoning about the *future* actions of a node, however, is a complex problem. This reasoning involves considering not only the current plans in the node's queue and making estimations about their durations and effects, but also what further information the node may receive (from another node or from its sensor) that could affect its activities. A plan may have associated with it some estimations as to duration and probability of completion, or even more specific information about how its execution could be affected by received information.

Our current implementation assumes that a node can make completely accurate short-term predictions about future activity based solely on the plan queue. We simulate this best-case scenario by allowing a node access

Environment	Type of nodes	Time
E1	KSI-based	49
	Plan-based	36
	Plan-based/MLC	26
	Optimal	23
E2	KSI-based	34
	Plan-based	26
	Plan-based/MLC	26
	Optimal	23
E3	KSI-based	66
	Plan-based	58
	Plan-based/MLC	38
	Optimal	35

Legend:
Time Earliest time at which a solution was found
MLC Meta-level Communication

Comparison of performance in multi-node environments. Plan-based nodes work consistently better than KSI-based nodes, resulting in improved performance. Meta-level communication can further improve performance by increasing coherence.

Table 1: Performance of Multi-node Networks.

to the abstracted blackboard and plan queue of another node. Discussion of more realistic scenarios where nodes must transmit this *meta-level* information as they transmit hypotheses, and must therefore reason about relevance, timeliness, and completeness, can be found elsewhere [6].

A node may use meta-level information to avoid redundancy. In developing a plan, a node can determine if the plan represents a redundant derivation of information that another node has either generated (present in the abstracted blackboard) or is in the process of generating (the top plan). By avoiding redundant activity, significant improvements in solution generation rate can result because less highly rated but potentially useful activities will be invoked earlier (rather than redundant invocation of highly rated activities). Hence, the experimental results in Table 1 indicate that network performance can be further improved by the exchange of meta-level information which allows individual nodes to make more coherent local decisions.

A. Coherent Communication

An important aspect of coherent network activity is that limited communication resources should be used intelligently to improve the global state of network problem solving. Flooding the bandwidth with partial results can cause both undesirable delays to important messages and unreasonable amounts of local processing as nodes incorporate the received information. On the other hand, if a node withholds certain partial results, network performance can degrade. It is therefore important that a node have a satisfactory view of both local and network problem solving in order to make coherent communication decisions.

We have developed a number of communication strategies that use the high-level view of node and network activity to guide a node in making these decisions [6]. These strategies extend the ideas first developed by Lesser and Erman [12]. When deciding about sending a partial result, the node might consider whether it will be improving upon that result in the future, and if so, whether by waiting and sending only the better version (conserving bandwidth) it can still fulfill its obligation to provide partial results in a timely manner. Furthermore, the exchange of meta-level information can allow a node to make inferences about how a particular transmission might affect network problem solving, and to decide when to repeat a message if the effects are not seen. In experimental studies, we have found that simple communication strategies that flood the bandwidth can significantly slow down the network, and that these problems become much more pronounced as we experiment with larger networks. For example, in the ten-node environment above (E3), a simple communication strategy results in 279 hypothesis transmissions between nodes. A more intelligent strategy reduces this number to 166 without adversely affecting the solution time. Therefore, indications are that coherence in communication decisions is an important area of study, and will become increasingly so as our environments continue to increase in size.

IV. SUMMARY AND IMPLICATIONS FOR DPSN RESEARCH

We have discussed our experiences in increasing the coherent behavior of the Distributed Vehicle Monitoring Test bed. We modified the blackboard problem solving architecture of the individual nodes to enhance their ability to make predictions about their future activities. Network coherence is increased by allowing a node to refine its organizational role based on these predictions. Exchanging the predictions permits a node to refine its view of the organisational roles of the other nodes.

Coherence is an integral part of distributed problem solving research. In contrast with others [11], we assume that we have only a limited number of highly sophisticated problem solving agents, and so, should coordinate them to make the most effective team possible. Because we assume that communication between agents is potentially slow and unreliable, we regard coordination that requires mutual agreement on contracts before action [1,4] to be insufficiently responsive to changing problem circumstances (indeed, mutual agreement might not even be possible [8]). To insure reliability, we cannot accept centralised coordination [1]. The unpredictable nature of the problem solving environment makes simple game theoretic models of agents unrealizable [15], while more complete models of agent beliefs [10] might require nodes to essentially duplicate each others reasoning.

Our view of distributed problem solving therefore stresses the importance of sophisticated local control which integrates object-level knowledge about the problem domain with meta-level knowledge about network coordination. Such control allows nodes to make rapid, intelligent local decisions based on changing problem characteristics without the overhead of conferring with each other. Coordination decisions are based on a high-level organizational view of individual node activity, so nodes need not have detailed models of the object-level problem solving activity of their compatriots. Dynamic improvements to this organisational view may be achieved with the exchange of meta-level messages which briefly convey high-level coordination information. In short, the nodes initiate their own activities and will take advantage of any local and network knowledge available to form the best "team" possible within the constraints of their environment.

We believe that implementation and experimentation are essential for learning about and understanding distributed problem solving. Our future plans include improving the representation of the state of a node, enhancing the mechanisms to recognise problem solving situations, and extending the plan structures to incorporate more information. Our preliminary experiments indicate that these developments should significantly improve the performance of distributed problem solving networks, and may also be useful in blackboard-based problem solving systems in general.

REFERENCES

[1] Stephanie Cammarata, David McArthur, and Randall Steeb
Strategies of cooperation in distributed problem solving.
In Proceedings of the Eighth International Conference on Artificial Intelligence, pages 767-770, August 1083.

[2] Daniel D. Corkill, Victor R. Lesser, and Eva Hudlicka.
Unifying data-directed and goal-directed control: An example and experiments.
In Proceedings of the Second National Conference on Artificial Intelligence, pages 143-147, August 1082.

[3] Daniel D. Corkill and Victor R. Lesser
The use of meta-level control for coordination in a distributed problem solving network.
In Proceedings of the Eighth International Conference on Artificial Intelligence, pages 748-756, August 1083.

[4] Randall Davb and Reid G. Smith
Negotiation as a metaphor for distributed problem solving.
Artificial Intelligence, 20(1083):63—100.

[6] Edmund H. Durfee, Daniel D. Corkill, and Victor R. Lesser
Distributing a distributed problem solving network simulator.
In Proceedings of the Fifth Real-time Systems Symposium, pages 237-246, December 1084.

[6] Edmund H. Durfee, Victor R. Lesser, and Daniel D. Corkill
Coherent Cooperation Among Communicating Problem Solvers.
Technical Report 85-15, Department of Computer and Information Science, University of Massachusetts, April 1086.

[7] Lee D. Erman, Frederick Hayes-Roth, Victor R. Lesser, D. Raj Reddy
The Hears ay-II speech understanding system: Integrating knowledge to resolve uncertainty.
Computing Surveys, 13(2):213—253, June 1080.

[8] Joseph Y. Halpern and Yoram Moses
Knowledge and common knowledge in a distributed environment.
In Proceedings of the Third ACM Conference on Principles of Distributed Computing.

[0] Barbara Hayes-Roth and Frederick Hayes-Roth
A cognitive model of planning.
Cognitive Science, 3:275-310, 1070.

[10] Kurt Konolige
A deductive model of belief.
In Proceedings of the Eighth International Conference on Artificial Intelligence, pages 377-381, August 1083.

[11] William A. Kornfeld and Carl E. Hewitt
The scientific community metaphor.
IEEE Transactions on Man, Systems, and Cybernetics, SMC-II(I):24-33, January 1081.

[12] Victor R. Lesser and Lee D. Erman.
An experiment in distributed interpretation.
IEEE Transactions on Computers, C-29(12):1144-1163, December 1080.

[13] Victor R. Lesser and Daniel D. Corkill.
Functionally accurate, cooperative distributed systems.
IEEE Transactions on Man, Systems, and Cybernetics, SMC-II(I):81-06, January 1081.

[14] Victor R. Lesser and Daniel D. Corkill.
The Distributed Vehicle Monitoring Testbed: A tool for investigating distributed problem solving networks.
AI Magasine, 4(3): 16-33, Fall 1083.

[15] Jeffrey S. Rosenschein and Michael R. Genesereth
Deals among rational agents.
Stanford Heuristic Programming Project Report No. HPP~ 84-44, December 1084.