A Cooperative Search Scheme for Dynamic Problems

Yasuhiko Kitamura, Zheng Bao Chauang, Shoji Tatsumi, and Takaaki Okumoto
Faculty of Engineering, Osaka City University
3-3-138 Sugimoto, Sumiyoshi-ku, Osaka, 558, JAPAN

S. Misbah Deen
DAKE Centre, University of Keele
Keele, Staffordshire, ST5 5BG, ENGLAND

Abstract **The importance of building a general framework for distributed problem solving is coming to be acknowledged. Distributed search is one of such frameworks and defined as to find a required path in a given graph by cooperation of multiple agents, each of which is able to search the graph partially. In this paper, we propose a new cooperative search scheme for dynamic problems where costs of links are changeable during search. To cope with the dynamic character, agents cooperate with each other by exchanging cost information that they keep. When the amount of exchanged information is large, the quality of solution is improved, but on the other hand it raises communication overhead. Therefore, it is significant to know how much information optimizes the performance. We developed a testbed that simulates a communication network and applied our scheme to the routing problem which can be viewed as a dynamic problem where the cost of a link is defined as its communication delay. We measured its performance according to the amount of the cost information exchanged.**

I. INTRODUCTION

Distributed problem solving (DPS) mainly studies how multiple agents cooperativelly solve a problem [3, 9, 7, 4]. Several experimental DPS systems and schemes so far have been proposed. Most of them, however, are implemented on specific architectures (e.g. blackboard system) and discussed on specific applications (e.g. sensor network), and it was difficult to make general discussions on their relations or comparisons among them. Hence, the importance of researches concerning formulations of DPS or generic problem solving schemes is being widely acknowledged [5].

As we can see problem solving has been formulated conventionally based on search [1], and various search algorithms are proposed based on heuristics [11], DPS can be formulated based on distributed search [10]. Assuming each agent searches a given graph partially, distributed path finding is one of distriubted search schems to find a path from a start node to a goal node by cooperations of multiple agents.

Diffusing search algorithm is such a distributed path finding algorithm. [10].

The search begins at an agent with the start node and it diffuses among agents until some agent finds the goal node. The problem dealt with in the scheme is a static problem where the cost of its links is fixed. On the other hand, we deal with dynamic problems where the cost is dynamically changeable in the cource of search. A typical example of this problem is a routing problem in communication networks. Assuming communication nodes and links are managed by agents in a distributed manner, we can view this problem as a distributed path finding problem to find an appropriate communication links from a node with a message to a node of its destination. When we view the communication delay of a link as its cost, as it is changeable according to the amount of incoming messages into the link, we can view this problem as a dynamic problem.

To deal with dynamic problems, we propose a cooperative scheme where agents exchange cost information and use it as estimates when required to find a path.

In this scheme, the more agents exchange the information, the better estimations which lead to find better paths are obtained. It, however, raises more communication overhead among agents and decreases the total performance. In other words, the performance improvement by cooperations is a trade-off against the communication overhead. Our question, therefore, is to know how much information the agents should exchange to make the total performance best and this is one of important general research issues of DPS[6]. We here introduce a device to adjust the amount of information by a parameter δ. Namely, when δ is low, more information is exchanged, but the estimations are more accurate. When it is high, the communication is supressed at the cost of the estimations. We made simulation experiments of a routing problem in a communication network to clarify how much δ is most appropriate depending on problems.

II. Distributed path finding problem

We here give a formulation of distributed path finding problem[10].

A. Graph

We give some basic terminology about graph theory for mathematical preparation.

A *graph* is denoted as a pair $< N, L >$ where $N(\neq \emptyset)$ is a set of *nodes* and $L(\subseteq N \times N)$ is a set of directed *links*. In this paper, we assume only finite graphs where the number of N is finite and $(n, n) \notin L$. A partial graph of $< N, L >$ is $< N', L' >$ which satisfies $N' \subseteq N, L' \subseteq L, L' \subseteq N' \times N'$.

For a given graph $< N, L >$, if $(n_i, n_j) \in L$, then n_j is said to be a *child* of n_i and n_i is said to be a *parent* of n_j. A sequence of nodes $(n_0, n_1, \ldots, n_m)(m \geq 1)$ is a *path* from n_0 to n_m if it satisfies $\forall k(0 \leq k \leq m-1) : \ell_{k,k+1}(= (n_k, n_{k+1})) \in L$, and m is the length of the path. When a *cost*, $c : L \rightarrow \mathbf{R}$ where $\mathbf{R}$ is the set of positive real numbers, is assign to each link, the cost of path $(n_0, n_1, \ldots, n_m)$ is defined as $\sum_{k=0}^{m-1} c(\ell_{k,k+1})$.

B. Problem and its solution

A *problem* is denoted as 3-tuple $< G, n_s, n_g >$ where $G =< N, L >$ is a graph and $n_s, n_g \in N$ are said to be *start node* and *goal node* respectively. For a given problem $< G, n_s, n_g >$, any path from n_s to n_g is a *solution* and the cost of the solution is given as that of the path. When a solution *sol* exists, if there is no other solution of which cost is smaller than that of *sol*, then *sol* is called *optimal*. In the course of problem solving, if the cost of links is fixed, the problem is said to be *static*, and otherwise, it is said to be *dynamic*.

C. Capability of agent

Distributed path finding problem is solved cooperatively by multiple *agents*. We call the set of agents *community* denoted by C. We assume an agent is a computational process and is capable of executing search algorithms and exchanging messages with other agents.

We assume also that *domain knowledge* $DK_a =< N_a, L_a >$ and *connection knowledge* $CK_a =< N_a, C >$ are available to each agent (a is the agent identifier). For a graph $G =< N, L >$ and an agent, the domain knowledge represents the partial graph which the agent can search and satisfies the followings.

- $\forall a \in C : N_a \subset N, L_a \subseteq N_a \times N_a, L_a \subset L$. In words, each agent can search a partial graph of the problem graph. A single agent, hence, may not be able to find a whole solution path.
- $N = \cup_{a \in C} N_a, L = \cup_{a \in C} L_a$. Any nodes and links consiting of the problem graph are included in the domain knowledge of some agent in the community. Hence, agents in the community can find a whole solution path by their cooperation.
- $\forall a, b \in C : a \neq b \Rightarrow L_a \cap L_b = \emptyset$. There is no duplicated links in the domain knowledge of agents. The community is a non-redundant system.

Connective knowledge represents relations among the domain knowledge and is denoted as a set of pairs of a node and an agent. If a node is included in domain knowledge of mutiple agents, it is called *connective node*. We say the node connects the agents. We have the following assumption concerning the connective knowledge.

- $\forall a \in C, \forall n \in N_a, \forall b \in C - \{a\} : n \in N_a \cap N_b \Leftrightarrow (n, b) \in CK_a$. In words, the connective knowledge of each agent includes all the pair of connective nodes and the connected agents.

D. Path finding algorithms

Non-distributed path finding problem is a classical problem and easily solvable by using traditional centralized search algorithms [8]. To solve the distributed version, which is our main concern, by those algorithms, it is needed to collect cost information managed by distributed agnets into a central agent. These centralized algorthms have drawbacks such as reliability that the breakdown of the central agent is fatal to the whole network and efficiency that the load and the communication are concentrated to the central agent.

On the other hand, the diffusing search algorithm[10] is a distributed version of the traditional search algorithm. In the algorithm, the search begins at an agent with the start node. The agent continues to search in the range of its domain knowledge, and if it reaches to a connective node, the agent requests the further search to the connected agent(s) by using its connetive knowledge. Once an agent finds a goal node, the search is terminated. In other words, the search diffuses from an agent with the start node among other agents. When the problem is static, it is possible to find an optimal solution by using this algorithm with the superiority to the centralized algorithm in its reliability and efficiency. In the case of dynamic problems, however, the diffusing search algorithm which finds a solution path by cooperation of multiple agents has the following shortcommings. Since the cost of links are variable dynamically in the course of search, the obtained optimal solution may have been obsolete. The diffusing search algorithm, therefore, is not appropriate for dynamic problems.

In IV.., we propose a cooperative search scheme which adaptable to dynamic problems by exchanging cost information among agents.

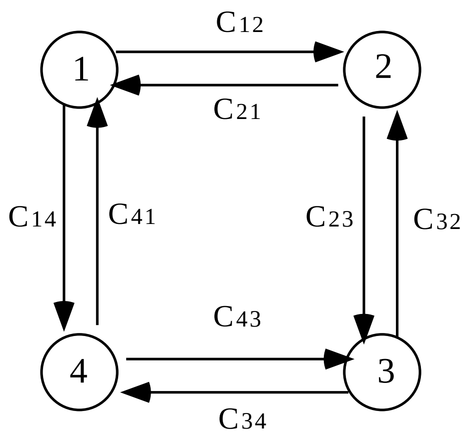

Figure 1: A Communication Network.

III. Routing Problem in Communication Networks

The routing problem in communication networks is one of dynamic ditributed path finding problem. A communication network can be viewed as a graph consisting of communication nodes and links. The cost of a communication link is given as its communication delay or its congestion. The problem is to find a path from a communication node with a message to a node of its destination with cost which is as less as possible[1].

Although an agent can manage multiple communication nodes and links, we assume that a signle agent manage only one node and links which is coming out of the node without loss of generality. Thus, in the following discussion, we often identifies a node with an agent which manages the node. By 'manage' we here mean to switch local message routing and to monitor the communication delay of local links. Multiple agnets thus need to cooperate to find a path of which the length is more than 2.

For example, Figure 1 shows a communication network consiting of 4 nodes and 8 links. Node 1 manages links C12 and C14, node 2 manages links C21 and C23, node 3 manages links C32 and C34, and node 4 manages links C41 and C43. When node 1 has a message destinated for node 3, there are two routes: one through node 2 and another through node4, it is required to know the correct communication delay of 4 links (C12,C23,C14,C43) to find the appropriate route. Since node 1 already knows the delays of links C12 and C14, it needs to cooperate with nodes 2 and 4 which knows the delays of links C23 and C43 respectively.

[1]Since this problem is a dynamic problem where the cost of link is changeable according to the amount of messages, it is difficult to define a optimal path.

Routing algorithms for communcation networks can be calssified into *nonadaptive* ones and *adaptive* ones [12]. Nonadaptive algorithms are not adative to changes of the topology or the communcation delay of communication networks, and are suitable for static problems. On the other hand, adaptive algorithms are suitable for dynamic problems and they are further classified into *global*, *local*, and *distirbuted* algorithms. Global algorithm gather the information distributed among nodes into a single node called RCC (Routing Control Center) which generates a routing table and distributes its copys among nodes. In this algorithm, since the prerequisite information is gathered into RCC, the routing is relatively correct and each node needs not to compute for its routing. On the other hand, RCC has to do a large amount of computation for the routing for the whole network and the breakdown of RCC is fatal.

Local algorithm is a way to route messages locally based on the information locally gatherd by each node. It is superior to global algorithm in reiability and load balancing, but has a drawback that the whole routing is inaccurate because of the locality of information.

Distributed algorithm is the combination of global algorithm and local algorithm and its routing is done locally by each node, but the nodes exchange the required information for the routing. In the next chapter, the proposed scheme is one of distributed algorithms. In the onventinal algorithms, nodes exhange the information synchronously with a fixed time interval. Our propsed algorithm exchanges the information asynchronously depending on the amount of communication delay of each link by using a threshold value and is more sensitive to the state of the network.

IV. Distributed path finding scheme for dynamic problems

In this chapter, we propose a distiributed path finding scheme for dynamic problems, using a routing problem in a communication network as an example.

A. Route selection

In our proposed scheme, the node with a message does not find a whole path to the node of its destination, but just finds a path to one of its adjacent nodes and send the message and commits to find the rest of the path to the adjacent node. This scheme thus is adaptive to changes of delay of links in the course of taking over the message.

Formally, we denote a node with a message as n_0, a node of its destination as n_g, adjacent nodes of n_0 as $n_1, \ldots, n_p$ where the message is transfered to n_g through one of those. When we define the delay through an adjacent node n_j as

$$c(n_j) = c(n_0, n_j) + c(n_j, n_g), \tag{1}$$

the selected node n_k should satisfy $c(n_k) = \min_{1 \le j \le p} c(n_j)$.

In this scheme, to find a better path is equivalent to find an more appropriate adjacent node. Under our assumptions, however, although a node knows the exact delay to its adajacent node, it has to use some estimation for the delay from the adjacent node to the destined node. In other words, in eq.(1), the first term is exact but we have to use an estimation $\tilde{c}(n_j, n_g)$ for the second term. We therefore use a scheme where nodes exchange delay information needed to calculate estimations.

B. Update of delay information

Nodes exchange delay information required to estimate communication delay. Formally, when node n_0 monitors a change of link delay $c(n_0, n_j)$ $(1 \le j \le p)$, it updates its delay information table and notifies the change to all the other nodes. The receivers also update their tables based on the received information.

When delay of links changes dynamically and frequently, nodes also need to exchange the information frequently to keep the estimations more accurate. On the other hand, this raises to increase the amount of communicatation and thus degrades the total performance. It can be viewed that the estimation quality is trade-off against the communication overhead and it becomes interesting to know how much communication makes the total performance optimal.

We here introduce a device to control communication to exchange the delay information. When a node notifies the delay, it records the value. We assume the initial value is 0. Only when the difference of change is more than δ, the node can notify the delay and update the record. In other words, only when $c(n_0, n_k) - c'(n_0, n_k) > \delta$, the node notifies and sets $c'(n_0, n_k) = c(n_0, n_k)$. To change the value δ, hence, enable to control the amount of communication. Namely, when δ is set to be lower, more messages are sent and the quality of estimation is improved. When it is higher, on the other hand, the quality is degraded but the amount of communication decreases.

V. Simulation experiments

We executed simulations on a communication network model in Figure 1. We assume each communication link is a queue. Messages coming in the link are queued and one message can get out of it and is transfered to the other node in a unit time. We define the delay of a link is the length of the corresponding queue.

We assume messages are generated at node 1, 2, and 4 and the destination of all is node 3. When node 3 receives a message, it sends back an ACK message. Messages generated at node 2 or 4 are regularly transfered through link C23 or C43 respectively, while messages at node 1 have two choices: through C12 and C23 or through C14 and C43. Node 1 selects the former when $d(C12) + d(C23) < d(C14) + d(C43)$ and the latter when $d(C12) + d(C23) > d(C14) + d(C43)$. Ties are broken rondomly. ($d(C)$ denotes the delay of link C or its estimate.) When node 3 sends back ACK messages, it follows the same routing process.

We assume messages generated at mode 1,2, and 4 follows a Poisson process [2] and we denote their averages as g_1, g_2, and g_4 respectively [2]. We change the arrival rates of nodes 2 and 4 by using eq.(2) where t is the current time.

$$P_1 + P_2 \sin(\omega t + \alpha). \tag{2}$$

Control messages with delay information are sent from nodes 2 and 4 to nodes 1 and 3 according to the parameter δ. Namely, node 2 and 4 sends the delay information of links C23 and C43 respectively to node 1 and that of links C21 and C41 respectively to node 3.

We executed a simulation to know how the performance changes according to the parameter δ. The arrival rates are $g_1 = 0.5$, $g_2 = 0.5 + 0.45\sin(2\pi t/1000)$, and $g_4 = 0.5 + 0.45\sin(2\pi t/1000 + \pi)$. In words, the communication between nodes 2 and 3 and between nodes 4 and 3 are congested in the interval of 1000 units time and their phases are opposite each other. Nodes 1 and 3, hence, have to change links according to the message arrivals at nodes 2 and 4.

To see how the performance changes according to δ without communication overhead to send delay information, we executed a simulation of an ideal case where nodes 2 and 4 can can send delay information without delay. The performance measure is the turnaround time, the communication time from when node 1 sends a message to when it receives a corresponding ACK message. We used the average of 100,000 messages. When the routing is good, this measure is low. We show the results in Figure 2(exclusive). As δ grows, it shows the performance is degraded.

When there is delay to send the delay information, in other words, normal messages and control messages are sent through the same link, we show the results in Figure 2(inclusive). When δ is low, the communication overhead dominates, while the deterioration of routing dominates, caused by the degraded quality of estimation, when δ is high. As δ grows, since the amount of control messages decreases, and the performance approaches that of the ideal case. In this simulation, the performance is optimal when $\delta = 5$.

[2] Time interval of messages follows an exponential distribution with the average $1/g$ unit time.

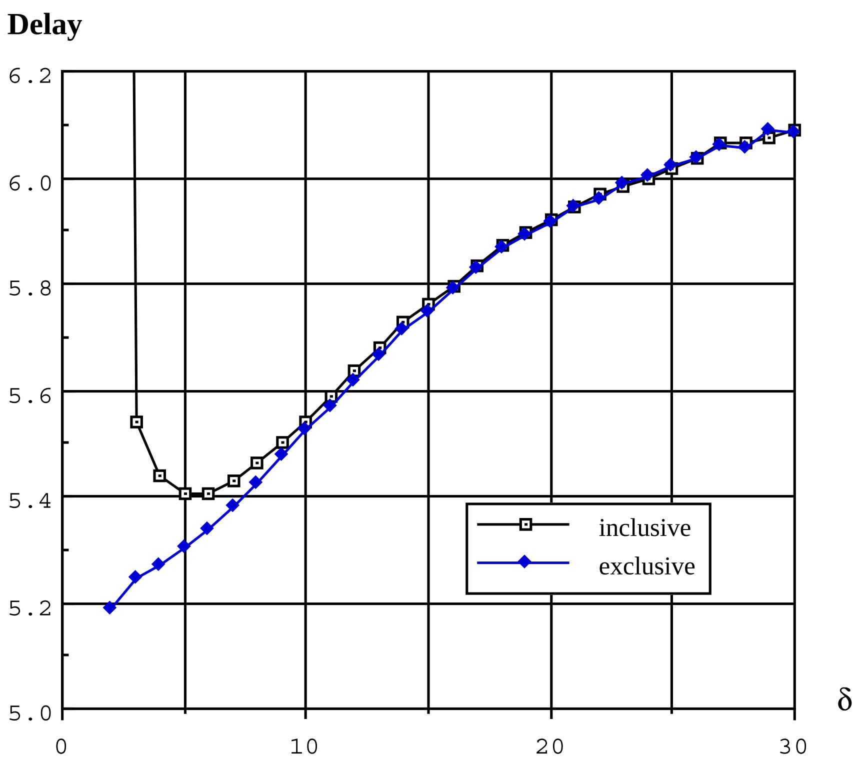

Figure 2: Simulation results.

VI. Summary

We proposed a cooprative path finding scheme for dynamic problems, using an example of routing in communication networks. To be adaptive to the dynamic change of link cost, a node with a message selects a routing only to one of adjacent nodes and commits the rest to the adjacent node. To select the adjacent node properly, nodes notify changes of link cost each other and use them as their estimations. Hence, to keep the estimations accurate, nodes need frequent exchange and that raise communication overhead which degrades the total performance. To cope with this problme, we introduced a device to control communication by using a parameter δ and showed through simulation experiments that there exists a amount of communication which optimizes the total performnce.

The followings are future studies.

- General evaluation: we showed through simulation experiments that there exists an optimal amount of communication for a specific case. The optimal amount may depend on the topology of network, the arrival rate of messages, the cycle of congestion, etc. We need a method to evaluate general cases.
- Automatic control of parameter δ: In general, the arrival ratio and the cycle of congestion change dynamically. We thus need, for each node, a device to control δ autonomously to optimize the total performance.
- Application to a large network: In this paper, we simulated a very small network, while if we apply our scheme to a large network, we need a more sophisticated device to cope with a large amount of communication. For example, when a path between two nodes is long, frequent exchange of detailed delay information between the two is inappropriate since most of the information is obsolete when it reaches to the opposite. We therefore can propose a scheme to change δ for each pair of nodes according to the length of the path between them.

References

[1] Ranan B. Banerji. *Artficial intelligence: a theoretical approach.* Elsevier North Holland, Inc., 1980.

[2] Dimitri Bertsekas and Robert Gallager. *Data Networks, Second Edition.* Prentice-Hall International, Inc., 1992.

[3] Alan H. Bond and Les Gasser, editors. *Readings in Distributed Artificial Intelligence.* Morgan Kaufman Publishers, Inc., San Mateo, California, 1988.

[4] Special section on distributed artficial intelligence. *IEEE Transactions on System, Man, and Cybernetics*, 21(6), November/December 1991.

[5] Edmund H. Durfee. The distributed artificial intelligence melting pot. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6):1301–1306, November/December 1991.

[6] Edmund H. Durfee, Victor R. Lesser, and Daniel D. Corkill. Coherent cooperation among communicating problem solvers. *IEEE Transactions on Computers*, C-36(11):1275–1291, November 1987.

[7] Les Gasser and Michael N. Huhns, editors. *Distributed Artificial Intelligence, Volume II.* Morgan Kaufman Publishers, Inc., San Mateo, California, 1989.

[8] E. Horowitz and S. Sahni. *Fundamentals of Computer Algorithms.* Computer Science Press, 1987.

[9] Michael N. Huhns, editor. *Distributed Artificial Intelligence.* Morgan Kaufman Publishers, Inc., Los Altos, California, 1987.

[10] Yasuhiko Kitamura and Takaaki Okumoto. Diffusing inference: An inference method for distributed problem solving. In S. M. Deen, editor, *Cooperating Knowledge Based Systems 1990*, pages 79–94. Springer-Verlag, 1991.

[11] Judea Pearl. *Heuristics.* Addison-Wesley, Reading, Massachusetts, 1984.

[12] Andrew S. Tanenbaum. *Computer Networks.* Prentice-Hall, 1988.