Communication Resource Sharing in Diffusing Inference

Yasuhiko Kitamura, Ken-ichi Teranishi, Shoji Tatsumi, and Takaaki Okumoto
Faculty of Engineering, Osaka City University
3-3-138 Sugimoto, Sumiyoshi-ku, Osaka 558, JAPAN
kitamura@info.osaka-cu.ac.jp

Abstract

The diffusing inference is a generic cooperative inference scheme which can be applied for heterogeneous distributed knowledge-bases (agents) to find a solution by cooperation of agents. A property of communication of the diffusing inference is characterized by nondeterministic communication by which we mean all the communication requests are not always required to be satisfied. In nondeterministic communication, there is a tradeoff between advantages such as solution quality or inference speed-up gained by sending many messages concurrently and disadvantages such as communication overhead. It is needed hence to send messages selectively to improve the total performance.

We propose, in this paper, local and global communication control schemes. In the local scheme, agents control their communication based on their local information. In the global scheme, the total amount communication in the whole system can be under control by using tokens which permit communication to agents. We evaluated these schemes by using a distributed maze problem from a standpoint of a tradeoff between inference speed-up and communication overhead. It is shown that the local scheme is ineffective once the inference spreads out among many agents and, on the other hand, the global scheme is superior to the local scheme and effective even when the communication cost is high.

1 Introduction

By advances in computer and communication technologies, our use of computers is changing from centralized use of a single computer to decentralized use of multiple computers. Currently we need our intervention to use distributed computers effectively, but in the future computers will have abilities to solve large-scale, distributed, and/or complex problems effectively cooperating with one another. Cooperation among computers, therefore, is one of key technologies for the next-generation computer systems built on wide-area and/or high-speed networks.

The *diffusing inference* [3] is a cooperative inference scheme to deduce a solution from distributed knowledge-bases, which can be viewed as a collection of agents. In the diffusing inference, an agent with a problem initiates an inference process. The agent solves the problem as far as it can. When it cannot solve the problem completely, it decomposes the problem into subproblems if it is possible, and sends a request message to one or more available agents for each of subproblems. The requested agents continue to solve or request the subproblems recursively likewise, so the inference processes diffuse among agents until a solution is found.

This kind of cooperative inference scheme is not only prerequisite for distributed knowledge bases but also an advantage of inference speed-up by making the computational load shared by multiple agents in cases the communication cost is small. On the other hand, the diffusing inference has a drawback because of its communication overhead. If we use this scheme naively, the amount of communication increases combinatorially as the inference spreads out among agents. Of course, this communication overhead can be viewed as a trade-off against advantages of the diffusing inference such as solution quality or inference speed-up.

In this paper, we propose and evaluate local and global communication control schemes to effectively share a common communication resource, which causes communication overhead in the diffusing inference, among agents and improve the total performance balancing inference speed-up and communication overhead.

Conventional AI problem solving involves nondeterministic procedures, for example 'trial-and-error', which may lead to combinatorial explosions of computation, and hence the inference control has been a major research issue. In the distributed environment, we face not only combinatorial explosions of computation but also those of communication. Considering the actual performance of computation and communication in current computer systems, coping with the latter comes to be another important research issue for building practical distributed knowledge-bases.

2 Diffusing Inference on Distributed Knowledge-Base

2.1 Distributed Knowledge-base

A *distributed knowledge-base system* is composed of multiple local knowledge-bases connected by a computer network. One way to build a distributed knowledge-base system is by extending a distributed database system [7], but this approach is possible only when all the local knowledge-bases are *homogeneous* in the sense that they use the same data model, for example the relational model. This proposition is not realistic when we make a large-scale distributed knowledge-base system by integrating local knowledge-bases which are designed independently.

For heterogeneous distributed knowledge-bases, we take an agent-oriented approach and show our model in Figure 1. An agent consists of a local knowledge-base, a cooperation module, and a communication module. The local knowledge-base of each agent can be designed independently and the cooperation module absorbs the difference and offers a common interface to other agents. For a global problem solving, agents cooperate through their cooperation module.

2.2 Problem and Its Decomposition

In this paper, we use a Prolog-like notation to represent problems and knowledge to solve problems. A *problem* is represented as a *goal*,

```
g(X)
```

with an arbitrary number (≥ 0) of arguments. An argument can be a constant or a variable. A problem `g(X)` is solved if a *fact* `g(a)`, which is unifiable with `g(X)`, is found. The fact is said to be a *solution* of `g(X)`. Facts may be stored in a local knowledge-base just like a database or may be obtained through inferential or computational processes like an expert system, depending on the design of local knowledge-base.

A problem can be AND- or OR- decomposed as specified by the following *rules*.

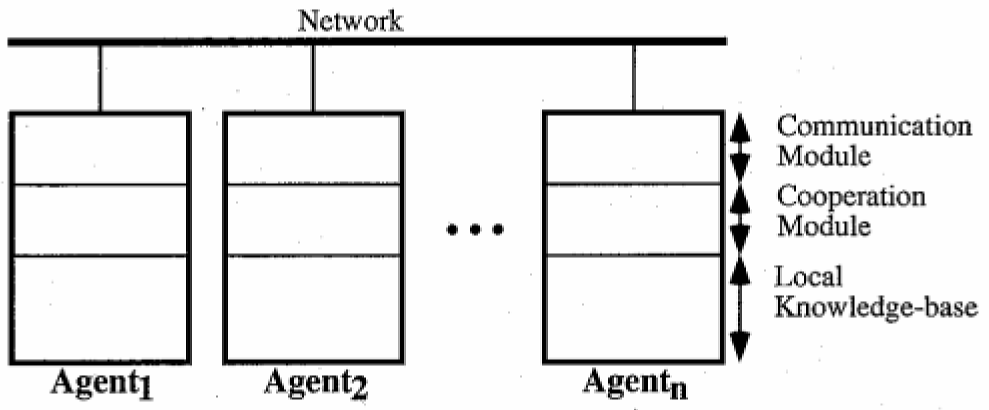

Figure 1: Model of heterogeneous knowledge-base system.

AND-decomposition rule: `g(X):-a(X),b(X).`
OR-decomposition rule: `g(X):-a(X);b(X).`

When a problem is AND-decomposed, it is required that all the sub-problems are solved. Moreover, if there are *constrains* among variables in the sub-problems, the constraints have to be satisfied. Namely, if there are identical variables in the sub-problems, they are required to be unifiable. When a problem is OR-decomposed, one sub-problem is sufficient to be solved.

2.3 Diffusing Inference

When an agent has a problem, it either (1) can find a solution in its local knowledge-base, (2) can decompose the problem by applying a rule, or (3) cannot solve the problem. In the distributed knowledge-base system, an agent (4) can cooperate with other agents by sending a request message to solve the problem when the agent cannot solve it by itself. Only when an agent with a problem cannot find a solution nor decompose the problem by itself and cannot find any other agent which can solve the problem, we say the agent cannot solve the problem. If agent α send a request message to agent β, we say α is a *parent agent* of β and β is a *child agent* of α.

We assume every agent knows the location of facts and facts are not redundant in distributed knowledge-bases for our convenience of the following discussion[1]. For any single problem, hence, every agent can specify a single agent at most which can handle the problem.

When an agent decomposes a problem into sub-problems including some are unsolvable by the agent, it requests other agents to solve them and composes a solution from sub-solutions by other agents. When a problem is OR-decomposed, the parent agent just selects one from sub-solutions by child agents. When a problem is AND-decomposed, however, there may be constraints among variables in the sub-problems and the composition of a solution is more complicated than that for OR-decomposition because the sub-problems may be solved independently by different child agents. For example, let us assume agent α has a AND-decomposition rule `g(X):-a(X),b(X).` and facts concerning `a` and agent β has facts concerning `b`. There is a constraint between the argument of `a` and that of `b`. As `a(X)` and `b(X)`are

[1]We assume each agent has a meta-database which stores the location information of facts or can use a protocol among agents to know it. [8]

solved independently by child agents α and β respectively, the agents need to coordinate to solve them satisfying the constraint.

The diffusing inference [3] is a cooperative inference scheme based on distributed search [1] to deduce a solution from distributed knowledge-bases (agents). An agent with an initial problem initiates an inference process. The agent solves the problem as far as it can. If it encounters an unsolvable problem, it sends a request message to an available agent if one exists. The descendant agent continues to solve the problem and/or request the sub-problem(s) likewise until a solution is found.

When an agent OR-decomposes a problem, the agent sends request messages for unsolvable sub-problems to solve them in parallel. When AND-decomposition, to deal with constraint satisfaction among variables in the sub-problems, the agent sends a request message to the agent which can handle the first unsolvable sub-problem.
For example, let us assume agent γ has a problem `g(X)` and an OR-decomposition rule `g(X):-a(X);b(X).`, agent α has facts concerning `a`, and agent β has facts concerning `b`. In this case, agent γ sends two individual request messages for sub-problems `a(X)` and `b(X)` to child agents α and β respectively to solve them in parallel. For a solution of `g(X)`, agent γ takes the earliest solution, `a(x1)` from α or `b(x2)` from β, if no other requirement is specified.

In the case where the above OR-decomposition rule is replaced by an AND-decomposition rule `g(X):-a(X),b(X).`, agent γ sends a request message for { `a(X),b(X)` } to agent α. If agent α finds a solution `a(x1)`, then it sends a request messages to verify `b(x1)` to agent β. If it is verified, agent α sends back a solution { `a(x1),b(x1)` } to agent γ.

The diffusing inference guarantees the completeness of inference. In other words, it guarantees to solve a problem if there exists a solution [3]. For OR-decomposition, it gains OR-parallelism by sharing efforts to find a solution among agents [4].

For AND-decomposition, there has been centralized and decentralized approaches to make sub-problems solved in parallel, satisfying constraints among variables. In a centralized approach such as distributed database [7], every child agent which solves sub-problems sends sub-solutions to the parent agent. The parent agent collects sub-solutions and composes a whole solution satisfying constraints. A drawback of this approach is that child agents do not use the constraint information, so they may send a number of useless sub-solutions which do not satisfy constraints.

In a decentralized approach such as distributed constraint satisfaction [10], agents exchange messages to coordinate their local problem solving to satisfy constraints. A large number of messages, however, may be required to be exchanged depending on how the variables are constrained. Generally speaking, the efficiency of distributed constraint satisfaction depends on the sequence of fixing variables and the appropriate sequence depends on the problem. In the diffusing search, the sequence is modifiable by changing the order of sub-problems in the AND-decomposition rule. The diffusing inference tries to reduce the overhead of distributed constraint satisfaction by sacrificing AND-parallelism. When a problem is AND-decomposed into independent sub-problems, it is easy to extend the diffusing inference to gain its AND-parallelism by defining independent sub-problem solving processes as individual diffusing inference processes.

3 Nondeterministic Communication in Diffusing Inference

There is a difference in communication property between conventional distributed processing systems and diffusing inference systems. In the former systems, all the communication requests are prerequisite to be satisfied. Once a message is generated, it must be transferred safely to its destination. We call this kind of communication *deterministic*. On the other hand, in the latter systems, all the communication requests are not always required to be satisfied, and we call this kind of communication *nondeterministic*. For example, let us assume a case that we make a travel plan. There are many travel agents which can help us to make itineraries and estimate their costs. The more travel agents we consult, the better plan we can get probably. On the other hand, if the travel agents take the consulting fee, the more we consult, the more we have to pay. What is worse, many of plans we get may look similar. We therefore decide the number of travel agents we consult considering the advantage of plans and the cost (money or time) to collect them.

We have a similar case in the diffusing inference, of which communication is nondeterministic, where an agent has a problem which is OR-decomposed into n sub-problems and cannot solve them without requesting the other agents. If the parent agent sends request messages for all the sub-problems at once as the original algorithm of diffusing inference, all the sub-problems will be processed in parallel at the cost of n messages and, consequently, the parent will be able to get the earliest solution from n agents or select the best one from many. On the other hand, if the purpose of the problem solving is just to find a single solution, n request messages are not always necessary to be sent. (In the worst case, n messages are still needed.) When the communication cost is not negligible, the more messages are sent, the more the network is congested and messages are delayed to be transferred. The total performance of diffusing inference therefore is affected by a trade-off between advantages such as inference speed-up and/or solution quality and disadvantages such as communication overhead, so each agent needs to control the amount of communication by selectively sending request messages according to their effects or importance. In a broad sense, since many agents are incorporated in a diffusing inference, the agents need to share the common communication resource effectively for one another.

4 Communication Resource Sharing

In the original diffusing inference algorithm, as soon as an agent encounters an unsolvable problem, it sends a request message. Hence, as the inference spreads out among many agents, the network is going to be congested delaying the message communication among agents. As we discussed, the request message communication in the diffusing inference is an instance of nondeterministic communication and we anticipate the performance of diffusing inference is affected by controlling the communication. In our new version, each agent controls to send request messages before they flow into the network as shown in Figure 2. Namely, each agent once stores generated request messages in its Message Queue and selectively sends them following the control strategy in NCCM.

We here define a term 'concurrent request'. We say request A is concurrent with request B if the request message for A is sent after the request message for B is sent and before the result message for B is received, or vice versa.

In this paper, we propose two control schemes; local and global. In the local scheme, each agent limits its maximum number of concurrent requests based on its local information only. In the global scheme, the total number of concurrent requests in the all agents are limited by using tokens.

4.1 Local Scheme

In the local scheme, each agent individually limits the maximum number of concurrent requests. We can further classify this scheme into the static and the dynamic ones. In the static local scheme, the maximum number is fixed, and in the dynamic local scheme, it dynamically varies according to the congestion of network.

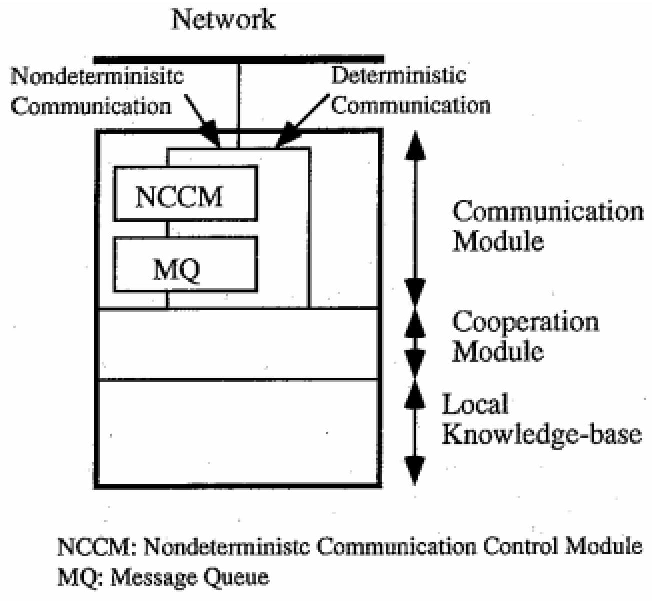
Network
Nondeterminisitc Communication
Deterministic Communication
NCCM
MQ
Communication Module
Cooperation Module
Local Knowledge-base
NCCM: Nondeterministc Communication Control Module
MQ: Message Queue

Figure 2: Communication module of agent.

Static Local Scheme: Each agent has a counter `RQ_C` which shows the number of concurrent requests. When an agent sends a request message, `RQ_C` is increased by 1, and when it receives a result message which shows a termination of a requesting problem solving, it is decreased by 1. In the static local scheme, we limit the maximum number of concurrent requests by setting a constant `rq_max` as the upper limit of `RQ_C`. Hence, when `RQ_C` $\geq$ `rq_max`, agents suspend to send request messages but stores them in their message queues. And, after when `RQ_C` $<$ `rq_max`, agents restart to send request messages.

Dynamic Local Scheme: In the dynamic local scheme, `rq_max` becomes a variable and each agent adjusts `rq_max` according to the congestion of network. Namely, when the network is congested, `rq_max` is set to be low, and when the network is not congested, it is set to be high. We use the turn-around time, which is the time interval between the dispatch of a message and the receipt of the corresponding acknowledged message in a low-level communication protocol, to estimate the congestion of network.

4.2 Global Scheme

The goal of global scheme is to control the total number of concurrent requests in the whole system. To embody this in a decentralized manner, we use tokens which permit concurrent requests. We limit the total number of concurrent requests by limiting the number of tokens which are distributed in the system. Initially tokens are with an agent which has an initial problem. When an agent sends a request message, it also sends one token at least with the message following the distribution rate $d (0 < d \leq 1)$. The number of tokens sent with a request message is defined as $\lceil d \cdot number\_of\_tokens \rceil$. Hence, the larger d is, the more tokens are sent to child agents.

As the diffusing inference spreads out, tokens are going to be distributed among agents. Concurrent requests of each agent are permitted while the number is lower than that of tokens which the agent possesses. Namely, rq_max is set to the number of tokens.

By introducing tokens, we need to study how tokens should be distributed among appropriate agents for they are effectively used. In principle, only agents with a token at least can forward the inference or, in other words, send request messages to its child agents, so tokens are flowing from the ancestor agent with the initial problem to its descendant agents. If tokens flow too fast, when a child agent fails to solve problems, it has to send back tokens to its parent agent to get new problems. If tokens flow too slow, the diffusing inference may not proceed forward and useless tokens may stagnate in ancestor agents. We therefore need to control the flow of tokens to distribute them appropriately balancing the progress of inference and the supply of new problems in case child agents fail to solve.

We here discuss the completeness of the diffusing inference in the global scheme. Since an agent can send request messages only when it has tokens, it is likely that a solution cannot be found even if it exists because tokens may not be distributed appropriately. Especially, tokens in inactive agents, which have no problem to solve, need to be assigned to other agents which need them. To deal with this, we adopt a simple way that, once an agent becomes inactive, tokens are distributed randomly to its parent and child agents. Hence, stochastically speaking, any agent which needs tokens will obtain one.

5 Experiments

In this section, we evaluate proposed communication control schemes by using a distributed maze problem.

5.1 Distributed Maze

Distributed maze problem is a problem to find a route from the entrance s_I to the exit s_G in a lattice maze as shown in Figure 3 by cooperation of multiple agents under an assumption that the maze is split and assigned to the agents. In the maze, there are obstacles located randomly and the hardness of the problem is defined by the obstacle rate. We use a 120×120 maze where its entrance and exit are $(1, 1)$ and $(120, 120)$ respectively. It is possible to move right, left, up, or down and not diagonally in the maze. The number of agents is 64 and the maze is split into 8×8 lattice, so each agent has the information of 15×15 lattice maze in its local knowledge-base.

Each agent uses A* algorithm [6] to search routes using a heuristic to evaluate a state (x, y) as $\hat{f}(x, y) = \sqrt{(x_g - x)^2 + (y_g - y)^2}$ where (x_g, y_g) is the location of the exit. We assume an agent consumes one unit time as it expands one state.

Agents are assumed to be connected with a single communication line like Ethernet, which is modeled as Figure 4. Messages generated in agents go into a single queue and a message comes out one by one in a time interval which is defined as the communication cost c. In this model, the more messages are generated among agents in a certain time interval, the longer the queue becomes and messages delay.

Initially, an agent which knows the location of the entrance starts to solve problem `route(sI,sG,R)` and tries to find a route as far as it can. When the search reaches a state `sM` on a border with its neighbor, it sends a request message for problem `route(sM,sG,R1)` to the neighbor agent. Likewise, the neighbor continues to solve the problem following the diffusing inference algorithm.

As the search process spreads out among agents, the efficiency of search is improved by making agents share the load. On the other hand, the communication overhead increases and degrading the performance. By using this testbed, we evaluate how efficiently agents share the common communication resource.

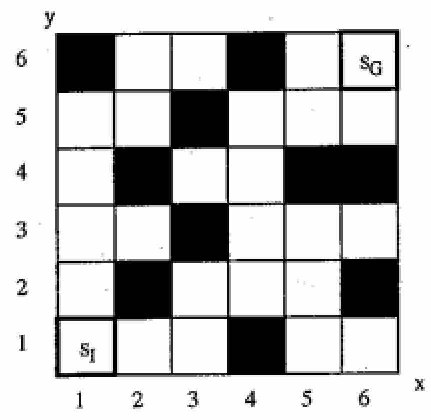

Figure 3: A lattice maze (6 × 6).

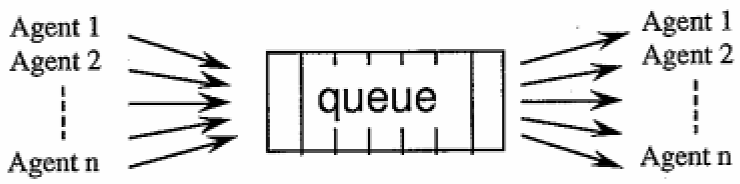

Figure 4: Model of communication line.

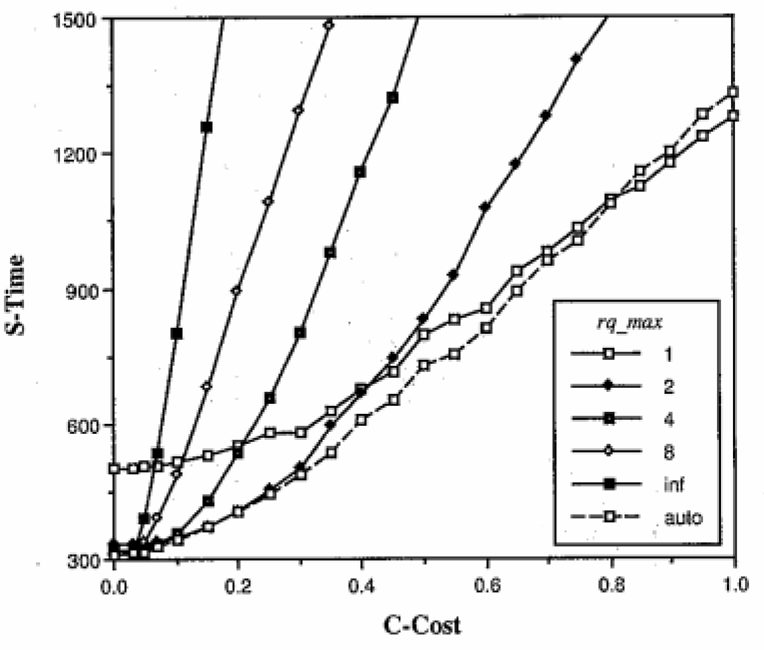

Figure 5: Performance of the local scheme.

5.2 Local Scheme

We show how the search time varies according to the communication cost in Figure 5. `rq_max` is set to 1,2,4,8, or infinity. The obstacle rate is 30%. In principle, when `rq_max` is high, the inference easily spreads out in parallel, but the network becomes saturated soon, so it shows good performance while the communication cost is low, but it goes worse as the cost increases.

Figure 5 (auto) shows the performance of the dynamic scheme which adjusts `rq_max` automatically according to the network congestion[2]. When the communication cost is low, it is superior to any other static schemes, but when the cost becomes high, it degrades the performance. A main reason is as follows. At the initial stage when the communication line is not congested, the inference spreads out promptly, but once many agents are involved in the inference, it is difficult to reduce the total amount of communication even if every agent has low `rq_max`. We therefore need a global scheme to limit the total amount of communication in the whole system.

5.3 Global Scheme

In the global scheme, an agent can send request messages while `RQ_C` $\leq$ `rq_max` and `rq_max` is set to the number of tokens which the agent possesses. Figure 6 shows the result when the total number of tokens varies from 5 to 100. The obstacle rate is set to 30% and the distribution rate is set to 50%. Compared with the local scheme (auto), the global scheme is superior if a large number of tokens are distributed when the communication cost is low. When it is high, the difference is outstanding if we limit the number of tokens to be low.

[2]We adjust `rq_max` based on results of the static scheme in an ad hoc manner.

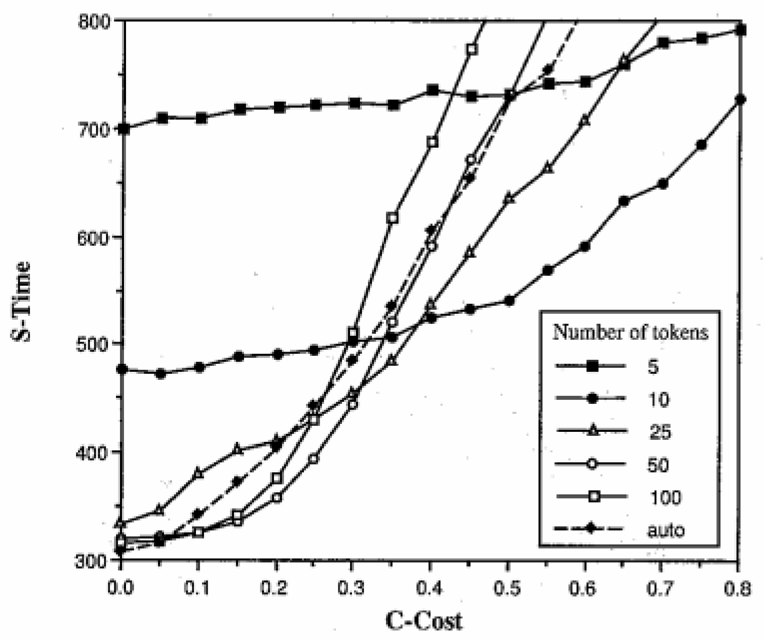

Figure 6: Performance of the global scheme when the number of tokens varies.

Figure 7 shows the result when the distribution rate varies. The obstacle rate and the number of tokens are set to 30 % and 50 respectively. When the communication cost is low, there is a tradeoff. If the distribution rate is too high, the inference proceeds fast but does not spread wide, hence, the performance is low because of its low parallelism. On the other hand, if the distribution rate is too low, the inference does not proceed forward. When the communication cost is high, it is better to set the distribution rate high to make the inference not spread too wide.

Figure 8 shows the effect of the distribution rate when the obstacle rate varies. The number of tokens and the communication cost is set to 50 and 0.2 respectively. When the obstacle rate is high, local searches fails easily. Hence, it is better to set the distribution rate low because parent agents can keep supplying requests to the child agents even if most of them fail in vain.

6 Related Works

Communication control schemes for deterministic communication has been studied in the field of computer networks as congestion control [9]. When a network is going to be congested, computers connected to the network interrupt messages going into it and restart when the congestion is relaxed. However, if we use this scheme for nondeterministic communication, communication among agents is just interrupted. On the contrary, messages which contribute to find solutions may be stagnated.

Another approach is to facilitate priority communication. We put a priority with each message as its importance. Since the network transfers messages according to their priority, it will be possible to improve the performance of the diffusing inference. However, this scheme is available only when priorities are rightly specified.

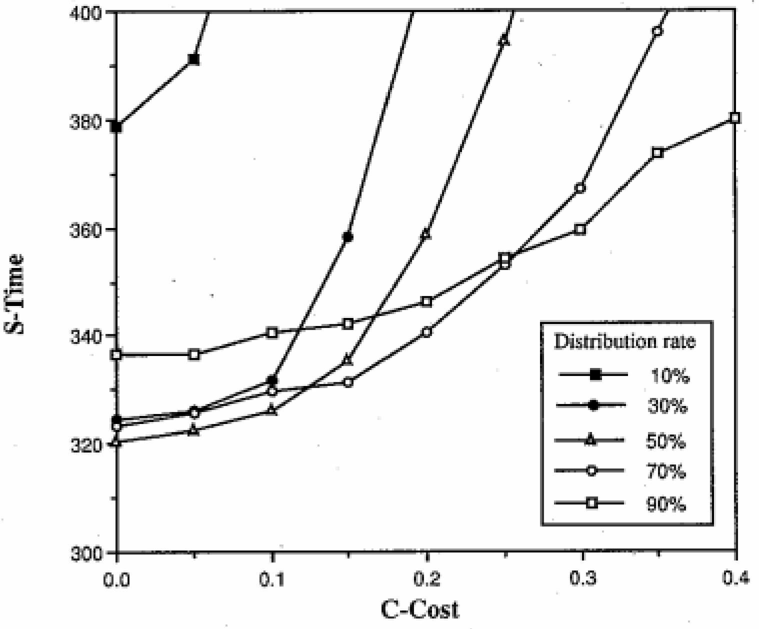

Figure 7: Performance of the global scheme when the distribution rate varies.

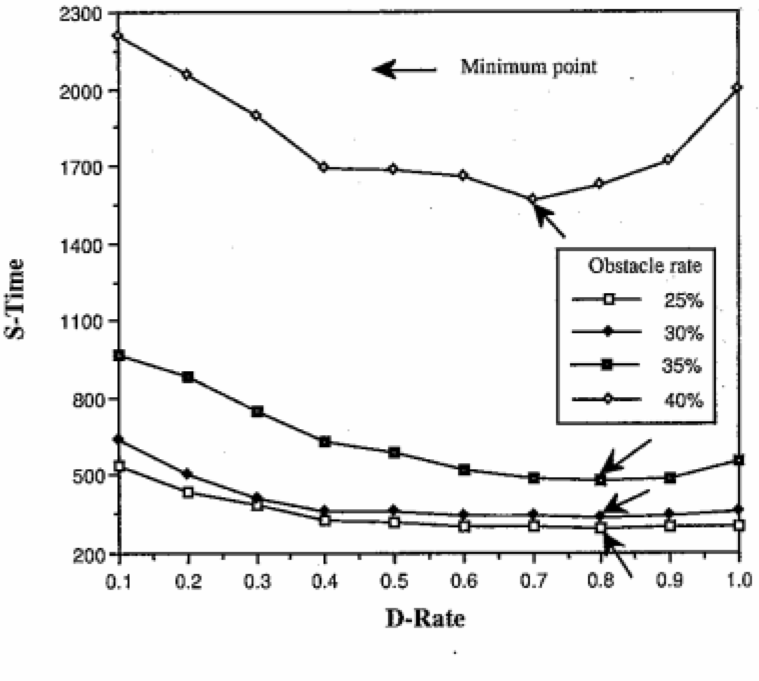

Figure 8: Performance of the global scheme when the obstacle rate varies.

Table 1: A classification of communication control schemes.

Level	Comm. property	Examples
High	Nondeterministic	FA/C [5], PGP [2]
Middle	Nondeterministic	Proposed schemes
Low	Deterministic	Flow control [9], priority communication

For nondeterministic communication, we need to control communication at a higher level than that for deterministic communication. In the field of DAI, some researchers are dealing with cooperation scheme considering communication overhead. For example, FA/C model [5] for hierarchal distributed problem solving exchange information among agent at a high level by abstracting a large amount of low-level information to reduce the amount of communication. However, this scheme is static and fixed when the system is designed. Partial Global Planning(PGP) [2] is a dynamic scheme to adapt to the change of environment and make problem solving plans considering the environment. However, as its communication control is done through an inference process of agent, it is not reactive enough to the change of communication network. And as the inference is based on its local information, it seems difficult to control communication appropriately in the global point of view as we discussed.

As shown in Table 1, we proposed communication control schemes at the middle level between the low level control which has bee studied in computer networks and the high level control which has been done in DAI. Our schemes therefore are available for nondeterministic communication and adaptive to the congestion of network.

7 Conclusions

We discussed communication control for nondeterministic communication in the diffusing inference, which is a generic cooperative inference scheme for heterogeneous distributed knowledge-bases, and proposed local and global control schemes. These schemes take an approach where messages are once stored in message queue and selectively sent following the communication control strategy.

We evaluated them through a simulation testbed of distributed maze problem from a standpoint of tradeoff between inference speed-up and communication overhead. In the local scheme, each agent controls communication locally, but it is not effective once the inferences spreads out among many agents. To control the total amount of communication in a decentralized manner, the global scheme uses tokens which permit communication. This scheme is shown effective even if the communication cost is high. In the global scheme, we need to continue to study how to distributed tokens to appropriate agents effectively.

References

[1] Edsger W. Dijkstra and C. S. Scholten. Termination detection for diffusing computations. *Information Processing Letters*, Vol. 11, No. 1, pp. 1–4, August 1980.

[2] Edmund H. Durfee and Victor R. Lesser. Partial global planning: A coordination framework for distributed hypothesis formation. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 21, No. 5, pp. 1167–1183, September/October 1991.

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

[4] Yasuhiko Kitamura, Shoji Tatsumi, and Takaaki Okumoto. Diffusing search scheme for distributed problem solving and its evaluation (in Japanese). *Transactions of Information Processing Society of Japan*, Vol. 35, No. 12, 1994.

[5] Victor R. Lesser and Daniel D. Corkill. Functionally accurate, cooperative distributed systems. *IEEE Transactions on System, Man an Cybernetics*, Vol. SMC-11, No. 1, pp. 81–96, January 1981.

[6] Nils J. Nilsson. *Problem Solving Methods in Artificial Intelligence*. McGraw-Hill, New York, 1971.

[7] M. Tamer Ozsu and Patrick Valduriez. *Principles of Distributed Database Systems*. Prentice-Hall International, 1991.

[8] Reid G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, Vol. C-29, No. 12, pp. 1104–1113, December 1980.

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

[10] Makoto Yokoo, Edmund H. Durfee, Toru Ishida, and Kazuhiro Kuwabara. Distributed constraint satisfaction for formalizing distributed problem solving. In *Proceedings of the 12th IEEE International Conference on Distributed Computing Systems*, pp. 614–621. 1992.