Organizational Strategies for Multiagent Real-time Search

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

Abstract

MultiAgent Real-Time A* (MARTA*) is a multi-agent version of Real-Time A* (RTA*) algorithm where multiple agents concurrently and autonomously search and move to find a solution. In this paper, we introduce two organizational strategies based on repulsion and attraction to MARTA*. Each agent observes the distances from others and moves as it becomes parted from or approaches others, in contrast it simply moves randomly in the original MARTA*. Through simulation experiments, we demonstrate repulsion and attraction are effective in both search time and solution quality for Maze and 15-puzzle respectively. However, the opposite combinations of strategy and problem degrade the performance. We finally discuss why the effectiveness of organizational strategy depends on the problem from a viewpoint of heuristic depression. Observed results suggest that there is a fertile research field which crosses over the traditional heuristic search and the organizational approach in multi-agent environment.

Introduction

Heuristic search is a fundamental method to deal with non-deterministic problem-solving which requires trial-and-error exploration of alternatives, and a number of search techniques, such as A* (Nilsson 1971) and Iterative Deepening A* (IDA*) (Korf 1985) algorithms to name a few, have been investigated (Pearl 1984). We can regard A* or IDA* as an off-line search algorithm which computes an entire solution path before executing the first step in the path. Korf (Korf 1990) proposed Real-Time A* (RTA*) as an on-line search algorithm which interleaves planning for a plausible next move and executing the move, motivated by the two-player game assumptions of limited search horizon and commitment to moves in constant time. RTA* do not find optimal solutions, but it can find semi-optimal solutions much quicker than traditional off-line search algorithms.

To improve solution quality of RTA*, Korf suggests increasing look-ahead horizon in the planning phase, but it makes the search time grow exponentially as the depth of look-ahead horizon increases. As an alternative way, Knight (Knight 1993) proposed MultiAgent Real-Time A* (MARTA*) algorithm to improve solution quality by increasing the number of agents engaged in the search. Multiple agents execute RTA* autonomously and concurrently to attack a single problem. When an agent chooses a next move at the planning phase and there are several candidates that look equally good, the agent chooses one randomly. By increasing the number of agents, we have more chances to find a better solution because more distinct paths toward goal states are explored. Since the computational complexity of MARTA* grows only linearly as the number of agents increases, Knight shows many reactive agents which look ahead shallowly are more effective for problems such as 15-puzzle than a few deliberative agents which look ahead deeply.

In the original MARTA* algorithm, however, each agent chooses its moves just independently and does not coordinate its moves with others. Multiple agents, therefore, may search the same path redundantly. In this paper, we have interest in how agents should be coordinated to find a better solution more rapidly in the context of MARTA* and propose two organizational strategies based on repulsion and attraction to make agents move in coordinated manners. Each agent observes the distances from others and moves as it becomes parted from or approaches others. Through simulation experiments, we demonstrate repulsion and attraction show better performance in both search time and solution quality for Maze and 15-puzzle respectively. However, interestingly the opposite combinations of strategy and problem degrade the performance. Therefore, we discuss why the effectiveness of organizational strategy depends on the problem from a viewpoint of heuristic depression (Ishida 1995). Observed results suggest that there is a fertile research field which crosses over the traditional heuristic search and the organizational approach in multi-agent environment.

Background

In this section, we give basic terms on graph and heuristic search for our following discussions.

Basic terms on graphs

A *graph* is represented 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*. We here assume $(n, n) \notin L$. In a graph $< N, L >$, if $(n_i, n_j) \in L$, then n_j is a *child* of n_i and n_i is a *parent* of n_j. A sequence of nodes $(n_0, n_1, \ldots, n_m)(m \geq 1)$ is called a *path* from n_0 to n_m if $\forall k(0 \leq k \leq m-1) : (n_k, n_{k+1}) \in L$, and m is called the *length* of the path. When a cost $c : L \rightarrow \mathbf{R}^+$, where $\mathbf{R}^+$ is the set of all positive real numbers, is attached to each link, the cost of path $(n_0, n_1, \ldots, n_m)$ is given as $\sum_{k=0}^{m-1} c(n_k, n_{k+1})$.

A *tree* is a graph in which every node except the *root* (node) has only one parent. A node with no child in a tree is called a *leaf*. The length from the root to a node n is called the depth to n.

Problem formulation

We can formulate non-deterministic problem solving with which AI has been dealing by using the state space representation as follows (Banerji 1980).

A problem is defined as $< S, O, s_I, G >$ where S is a non-empty set of *states*, $O(\subseteq S \times S)$ is a set of *operators*, each of which maps a state to another, $s_I(\in S)$ is the *initial state*, and $G(\subseteq S)$ is a set of *goal states*. $< S, O >$ constitutes a *state space* graph.

A *solution* is defined as any path $(s_0, s_1, \ldots, s_n)$ from the initial state s_I to a goal state $s_G \in G$ where $s_0 = s_I$ and $s_n = s_G$. When a cost (> 0) is attached to each operator, the cost of solution is given as that of solution path. If there exists a solution with cost c_{min} and no other solution with cost less than c_{min}, then the solution is *optimal*.

Heuristic search

The most basic operation of state space search is generating a child state by applying an operator to a state. Especially, if we generate all the child states of a state, then we say the parent state is *expanded*. We begin a graph search by expanding the initial state and repeatedly expand generated states until a goal state is generated.

Although there are various search techniques such as depth-first or breadth-first characterized by the order of generated states, these brute force algorithms are not tractable in fact for problems whose state space is large because of its combinatorial explosion of computation. Hence, to improve the search efficiency, we introduce a state evaluation function to prioritize states to be expanded. In general, the following evaluation function $f(s)$ is widely used as in A* (Nilsson 1971).

$$\hat{f}(s) = \hat{g}(s) + \hat{h}(s),$$

where $\hat{g}(s)$ is the actual cost of reaching state s from the initial state, and $\hat{h}(s)$ is the estimated cost of reaching the goal state from state s. A* algorithm has a property that it always finds an optimal solution if $\hat{f}(s)$ never overestimates the actual solution cost. A major drawback of A* algorithm is that it requires exponential time and space.

Real-time search

We can regard A* as an off-line search algorithm which computes an entire solution path before executing the first step in the path. Korf (Korf 1990) proposed Real-Time A* (RTA*) algorithm as an on-line search algorithm which interleaves planning for a plausible next move and executing the move as follows.

Step1 [Initializing] Set $s_x = s_I$.

Step2 [Expansion] Expand s_x and let $C(s_x)$ to be the set of child states.

Step3 [Termination?] If there exists a goal state in $C(s_x)$, then move to the goal state and quit.

Step4 [Look-ahead search] For all $s_y \in C(s_x)$, calculate $\hat{f}(s_x, s_y) = c(s_x, s_y) + \hat{f}(s_y)$ which is the estimated cost from s_x to the goal state through s_y. $\hat{f}(s_y)$ is calculated from a look-ahead search of depth d from s_y as follows.

$$\hat{f}(s_y) = \min_{s_w \in W(s_y, d)} [\hat{c}(s_y, s_w) + \hat{h}(s_w)],$$

where $W(s_y, d)$ is the set of leaves of the look-ahead search tree and $\hat{c}(s_y, s_w)$ is the actual cost from s_y to s_w known at the stage.

Step5 [Choice] Choose the best child state s'_y with $\hat{f}(s_x, s'_y)(= \min_{s_y \in C(s_x)} \hat{f}(s_x, s_y))$. Ties are broken randomly.

Step6 [Estimation update] Update $\hat{h}(s_x)$ to be $\hat{f}(s_x, s''_y)(= \min_{s_y \in C(s_x) - \{s'_y\}} \hat{f}(s_x, s_y))$ which is the estimated cost of the second-best child. If there is no second-best estimation, let $\hat{h}(s_x) = \infty$.

Step7 [Move] Set $s_x = s'_y$.

Step8 Go to Step2.

Since RTA* does not use $\hat{g}(\cdot)$ which is the actual cost from the initial state but only $\hat{h}(\cdot)$ which is the estimated cost to the goal state, it may fall into an infinite loop in which the same state may be visited infinite times because the evaluation function $\hat{f}(\cdot)$ does not reflect the search history. In fact, RTA* guarantees not to fall into an infinite loop as $\hat{h}(\cdot)$ is updated as to be increased monotonically in Step 6, and to find a solution if there exists one at least. In RTA* in which a single agent is engaged in search, updating $\hat{h}(\cdot)$ with the second-best estimated cost hinders itself from visiting the same state again. On the other hand, the estimated cost is likely to overestimate the actual one.

RTA* do not find optimal solutions, but it can find solutions much quicker than traditional off-line search algorithms. Korf suggests a remedy to improve solution quality in RTA*. It is by increasing look-ahead horizon d, but there is a drawback that the search time grows exponentially as d increases. Korf proposed a learning version of RTA* also. In the Learning RTA* (LRTA*) algorithm, $\hat{h}(s_x)$ is updated with the best estimation $\hat{f}(s_x, s'_y)$ for not overestimating the actual cost. Henceforth, LRTA* guarantees that $\hat{h}(\cdot)$ will eventually converges to the optimal value, but it takes more time than RTA* because it needs to revisit the same state more times than RTA*.

Multiagent real-time search

As an alternative to improve solution quality in RTA*, Knight (Knight 1993) proposed MultiAgent Real-Time A* (MARTA*) algorithm in which multiple agents autonomously and concurrently execute RTA* where the look-ahead horizon is set to be 1.

MARTA* has two advantageous properties.

Discovering effect The more agents are engaged in a search, the more distinct paths toward goal states are discovered, and this leads to improving solution quality. Even if some agents get stuck, others are still able to continue to search.

Learning effect The more agents are engaged in a search, the more actively agents update estimated costs, and this also leads to improving solution quality.

While the search time of RTA* grows exponentially as the look-ahead horizon increases, that of MARTA* grows only linearly as the number of agents increases. Moreover, it is also advantageous that MARTA* can be implemented on parallel processors naturally.

However, there are two flaws in MARTA* concerning updating estimated costs and coordinating agents' move. When an agent updates $\hat{h}(\cdot)$ in MARTA*, it uses RTA* method in which $\hat{h}(\cdot)$ is updated with the second-best estimation which may be overestimating the state. Since multiple agents share $\hat{h}(\cdot)$ in MARTA*, overestimated $\hat{h}(\cdot)$ hinders other agents from visiting the state. On the other hand, LRTA* method in which $\hat{h}(\cdot)$ is updated with the best estimation does not overestimate, but increases the search time because agents tend to visit the same state again.

In this paper, we devise a hybrid updating method by combining RTA* and LRTA* methods. Namely, we use two sorts of estimated costs; common $\hat{h}_G(\cdot)$ shared by all the agents and local $\hat{h}_L(\cdot)$ for each agent. $\hat{h}_G(\cdot)$ is updated by using the LRTA* method not to overestimate, and $\hat{h}_L(\cdot)$ is locally updated by using the RTA* method for the agent to refrain from visiting the same state again. Our new algorithm is obtained from modifying Step 4 and Step 6 in RTA* as follows.

Step4 [Look-ahead search] For all $s_y \in C(s_x)$, calculate $\hat{f}(s_x, s_y) = c(s_x, s_y) + \hat{h}(s_y)$ which is the estimation of optimal cost from s_x to a goal state through s_y. $\hat{h}(s_y)$ is calculated as follows.

$$\hat{h}(s_y) = \begin{cases} \hat{h}_G(s_y) & \text{if } s_y \text{ has not been visited,} \\ \hat{h}_L(s_y) & \text{if } s_y \text{ has been visited.} \end{cases}$$

Step6 [Estimation update] Update $\hat{h}_G(s_x)$ to be $\hat{f}(s_x, s'_y)$ which is the best estimation and $\hat{h}_L(s_x)$ to be $\hat{f}(s_x, s''_y)(= \min_{s_y \in C(s_x) - \{s'_y\}} \hat{f}(s_x, s_y))$ which is the second-best estimation. If there is no second-best estimation, let $\hat{h}_L(s_x) = \infty$.

This hybrid updating method shows improvement in solution quality comparing with the original MARTA* in our preliminary experiment.

The second flaw is concerning coordination of agents' moves. In the original MARTA*, when an agent meets multiple child states that look equally good, it selects one randomly, but this tends to redundant search and hence degrades the performance. In the next section, we propose two organizational strategies based on repulsion and attraction to make agents move in a coordinated manner.

Organizational strategies for MARTA*

In the original MARTA*, agents do not move in a coordinated way but just randomly. We expect coordinated moving agents according to the problem would show better performance than random moving agents. We, in this paper, propose two organizational strategies based on repulsion and attraction, to coordinate agents' move. Repulsion is expected to strengthen the discovering effect by scattering agents in a wider search space. In contrast, attraction is expected to strengthen the learning effect by making agents update estimated costs in a smaller search space more actively.

Repulsion

Each agent observes the location of others, and moves in a direction such that it repels others when it has candidates to move that look equally good.

We here define the term *adjacency* of an agent which represents a locational relation of the agent with other agents. Adjacency $\delta(i, s_i)$ of agent i at state s_i is calculated as follows.

$$\delta(i, s_i) = \min_{j \in A - \{i\}} \hat{d}(s_i, s_j),$$

where A is the set of agents engaged in MARTA*, s_j is the state where agent j is located, $\hat{d}(s_i, s_j)$ is the estimated distance from s_i to s_j. Namely, adjacency represents the distance to the nearest agent.

We next define *repulsive range* R as the factor to decide whether the agent repulses others or not. Repulsive range $R(s_i)$ for agent i at state s_i is calculated

as follows.

$$R(s_i) = \frac{\alpha \times \hat{h}(s_i)}{\hat{h}(s_I)},$$

where s_I is the initial state. The repulsive range is largest when the agent is located at the initial state, then reduces as it is approaching the goal state. Hence, agents strongly repulse each other at the initial stage of search, but they are converging on the goal state since the repulsive range becomes smaller as they approach the goal state. By changing α, we can change the size of repulsive range.

Using adjacency δ and repulsive range R, we now modify Step 5 of MARTA* as follows.

Step5 [Choice] Agent i chooses s'_y from the candidate set $M_i = \{s_y | \hat{f}(s_i, s_y) = \min_{s_z \in C(s_i)}[c(s_i, s_z) + \hat{h}(s_z)]\}$ as follows.

$$\begin{cases} \text{Choose } s'_y & \text{as it satisfies} \\ & \delta(i, s'_y) = \max_{s_y \in M} \delta(i, s_y) \\ & \text{if } \forall s_y \in M : \delta(i, s_y) < R(s_i). \\ \text{Choose } s'_y & \text{as it satisfies } \delta(i, s'_y) \geq R(s_i) \\ & \text{otherwise.} \end{cases}$$

Ties are broken randomly.

In words, if all candidates are within the repulsive range, choose one with a maximum adjacency. Otherwise, choose one from outside of the repulsive range randomly.

Attraction

Each agent observes the location of others, and moves in a direction such as it approaches others when it has multiple candidates to move that look equally good.

We here define a term *isolation* Δ of an agent. Isolation $\Delta(i, s_i)$ of agent i at state s_i is calculated as follows.

$$\Delta(i, s_i) = \max_{j \in A - \{i\}} \hat{d}(s_i, s_j).$$

We here use *attraction range* G for an agent to decide whether it approaches other agents or not. When G is large, agents are scattered, and when small, agents move crowdedly.

Step 5 of MARTA* is modified as follows.

Step5 [Choice] Agent i selects s'_y from the candidate set $M_i = \{s_y | \hat{f}(s_i, s_y) = \min_{s_z \in C(s_i)}[c(s_i, s_z) + \hat{h}(s_z)]\}$ as follows.

$$\begin{cases} \text{Choose } s'_y & \text{as it satisfies} \\ & \Delta(i, s'_y) = \min_{s_y \in M} \Delta(i, s_y) \\ & \text{if } \forall s_y \in M : \Delta(i, s_y) > G. \\ \text{Choose } s'_y & \text{as it satisfies } \Delta(i, s'_y) \leq G \\ & \text{otherwise.} \end{cases}$$

Ties are broken randomly.

In words, if all the candidates are located outside of the attraction range, the agent selects one with a minimum isolation. Otherwise, select one in the attraction range randomly.

It is easy to combine two strategies. When using the hybrid strategy, agents are located in a doughnut-area defined by R and G.

Overhead

In the original MARTA*, as each agent moves independently, there is no overhead for coordination. In our organizational approach, we need to spend additional computational cost to coordinate agents' moves. For each agent, when it decides the next state to move, it refers to the distances to all the other agents for calculating adjacency and isolation, so the total computational cost for each turn becomes $O(\mathcal{A}^2)$, where $\mathcal{A}$ is the number of agents engaged in MARTA*.

Experimental evaluation

Evaluation problems

To evaluate proposed organizational strategies, we solved Maze and 15-puzzle problems by using MARTA* algorithm.

Maze problem: This is a problem to find a path from the entrance to the exit in a rectangular grid problem space with randomly positioned obstacles (Ishida & Korf 1991). Each agent can move in the horizontal or vertical direction, but not in the diagonal direction. We assume an agent takes a single cost for a single move. We use the 120×120 grid and positioned obstacles at a ratio of 40%. The entrance and the exit are located at $(1, 1)$ and $(120, 120)$ respectively. We use the Euclidean distance to the goal state for the initial estimated cost of each state and adjacency and isolation of each agent. We execute 100 trials of MARTA* for 100 randomly generated problems with solutions.

15-puzzle problem: The 15-puzzle consists of a 4×4 square frame containing 15 numbered square tiles and a single blank. The legal operators slide any tile horizontally or vertically adjacent to the blank into the blank position. The task is to rearrange the tiles from some random initial configuration into a particular desired goal configuration. We assume an agent takes a single cost for a single move of tile. We use the Manhattan distance for the initial estimated cost of each state and adjacency and isolation of each agent. It is computed by counting, for each tile not in its goal position, the number of moves along the grid it is away from its goal position, and summing these values over all tiles. We execute 100 trials of MARTA* for 9 problems selected from 100 problems presented in (Korf 1985). [1]

[1] We selected 3 difficult , 3 middle, and 3 easy problems.

Results

We solve Maze and 15-puzzle problems by using MARTA* simulating agents as if they run concurrently on parallel processors with a shared memory. We assume the accessing cost to the shared memory is negligibly small. We also do not count computational costs taken for look-ahead search and organizational strategies and normalize the time taken for a single move as a unit time. Then, we measure search time and solution length. The search time is the time taken by the first agent which reaches a goal state and the solution length is obtained by deleting cycles from the found solution path.

For evaluation of repulsion, we set the attraction range to be infinity ($G = \infty$), then we varied the parameter α. When $\alpha = 0$, it is equivalent to the original MARTA*. The averaged results of Maze and 15-puzzle are shown in Figure 1 and Figure 2 respectively. X-axis represents the number of agents. The results of the experiments show repulsion is effective for Maze, but not for 15-puzzle.

For attraction, we set the repulsion range to be 0 ($\alpha = 0$) and varied the attraction range G. When $G = \infty$, it is equivalent to the original MARTA*. We show the averaged results of Maze and 15-puzzle in Figure 3 and Figure 4 respectively. The results of the experiments show that attraction is effective for 15-puzzle, but not for Maze. This result makes a sharp contrast to that of repulsion.

Discussion

We now explain why the two organizational strategies showed contrasting results by using the concept of *heuristic depression* (Ishida 1995). Heuristic depression is a set of connected states whose estimated cost are less than or equal to those of the set of immediate and completely surrounding states. Each agent which performs MARTA* simply selects a state to move with the least estimated cost from candidates, so it easily moves to the bottom state of a heuristic depression. Once an agent falls into a heuristic depression, it cannot escape from it without filling the depression, namely updating the estimated cost $\hat{h}(\cdot)$ of every states in the depression until they are equal to those of the surrounding states.

Here we estimate how widely and deeply heuristic depressions are distributed in Maze and 15-puzzle by calculating the difference between the initial estimated cost and the actual cost of states through which agents move on the way to the goal state as follows.

1. Execute a single agent RTA* and record the states which the agent visited and their initial estimated cost.
2. Perform the breadth-first search from the goal state and calculate the actual costs of the recorded states.
3. Calculate the difference between the actual cost and the initial estimated cost of the recorded states.

As the size of search space of 15-puzzle is huge, we used 8-puzzle, which seems to have similar heuristic depressions, instead of 15-puzzle.

The measurement results of Maze and 8-puzzle are shown in Figure 5(a) and (b) respectively. The X-axis represents the difference between the actual cost and the initial estimated cost ($h(s) - \hat{h}(s)$) and the Y-axis represents the percentage of states. In Maze, the difference distributes widely from 0 to 500. The ratio of states with the same value is less than 1%. In contrast, in 8-puzzle, the difference distributes densely from 0 to 20. Comparing these results, we can suppose that deep heuristic depressions are scattered in Maze more than in 8-puzzle.

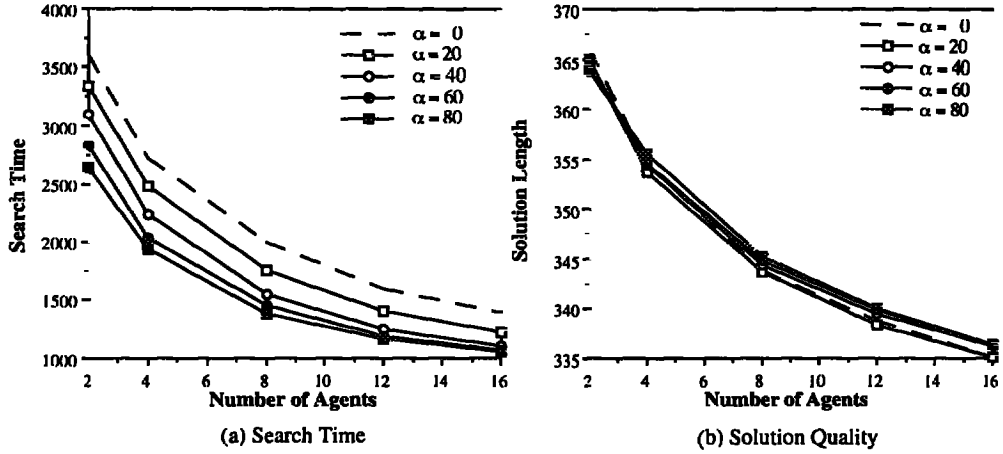
In Maze, as heuristic depressions are spread between the initial state and walls consisting of obstacles, there are deep depressions spotted in the state space. On the other hand, in 8-puzzle or 15-puzzle, shallow depressions are distributed ubiquitously. Therefore, attracting agents as a whole easily fall into deep depressions in Maze. In contrast, as repulsing agents move keeping some distance with each other, even if some fall into a depression, the rest can continue to search toward the goal state. Hence, for Maze, repulsing agents are effective.

In 15-puzzle, there are many shallow depressions. In such a state space, it is better for agents to move densely rather than separately to fill depressions collaboratively, namely to update estimated costs in collaboration with others. For 15-puzzle, attracting agents are effective.

Conclusion

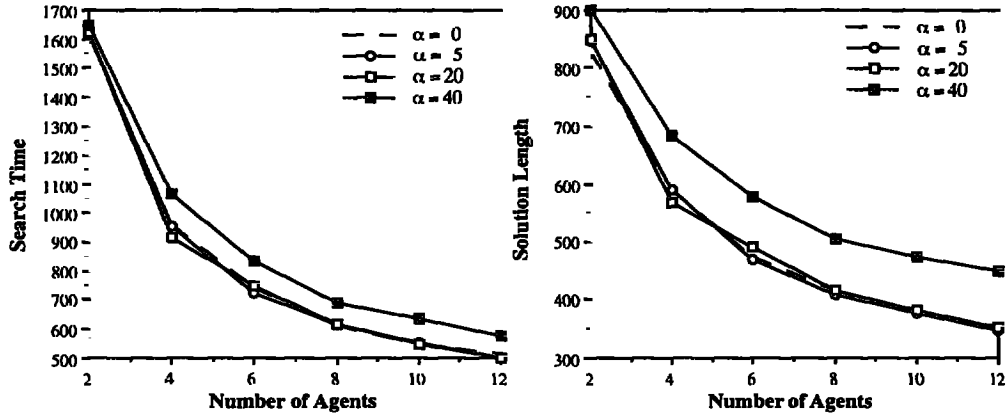
We proposed and evaluated two organizational strategies based on repulsion and attraction for MARTA* algorithm. Each agent engaged in the search observes the location of others and makes a move as it separates from or approach others. MARTA* has two advantages: discovering effect and learning effect, and they are strengthened by repulsion and attraction respectively. For their evaluations, we used Maze and 15-puzzle which are contrasting from a viewpoint of heuristic depression. For Maze in which deep depressions are spotted, repulsion showed a good performance, and for 15-puzzle in which shallow depressions are distributed ubiquitously, attraction showed a good performance. In conclusion, if we use an appropriate organizational strategy according to the characteristic of problem, we can get better performance than using random moving agents.

In this paper, we just showed a relation between appropriate organizations and heuristic depressions in Maze and 15-puzzle problems, so application of our organizational approach to other problems and performance evaluation in detail including the overhead of coordination remain as our future study. However, observed results suggest that there seems to be a fertile research field which crosses over the traditional heuris-



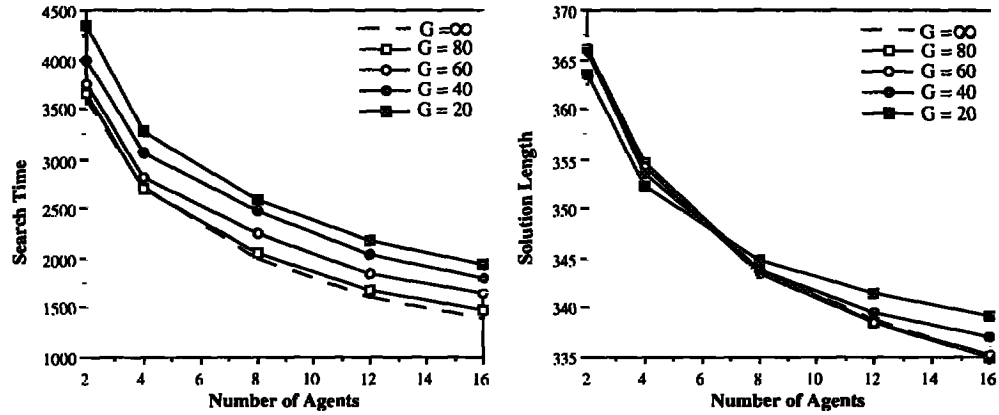
(a) Search Time (b) Solution Quality

Figure 1: The effect of repulsion on Maze.



(a) Search Time (b) Solution Quality

Figure 2: The effect of repulsion on 15-puzzle.



(a) Search Time (b) Solution Quality

Figure 3: The effect of attraction on Maze.

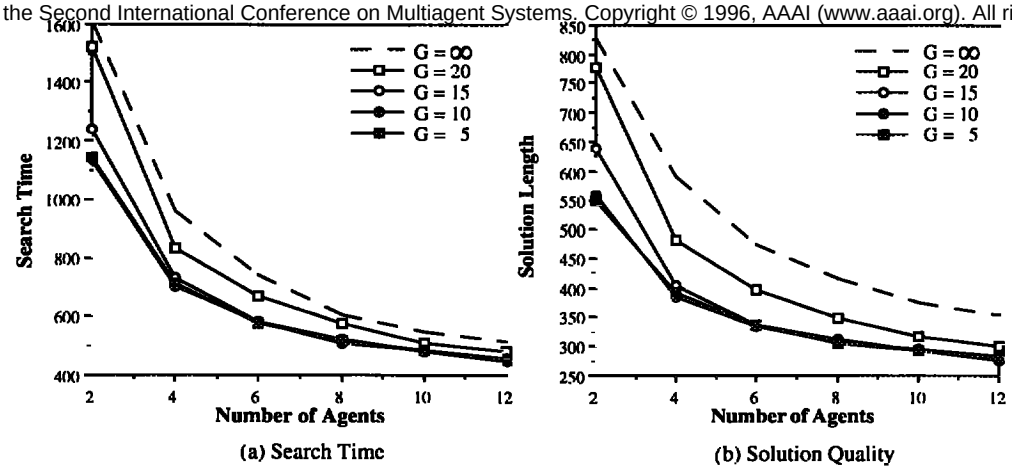


Figure 4: The effect of attraction on 15-puzzle.

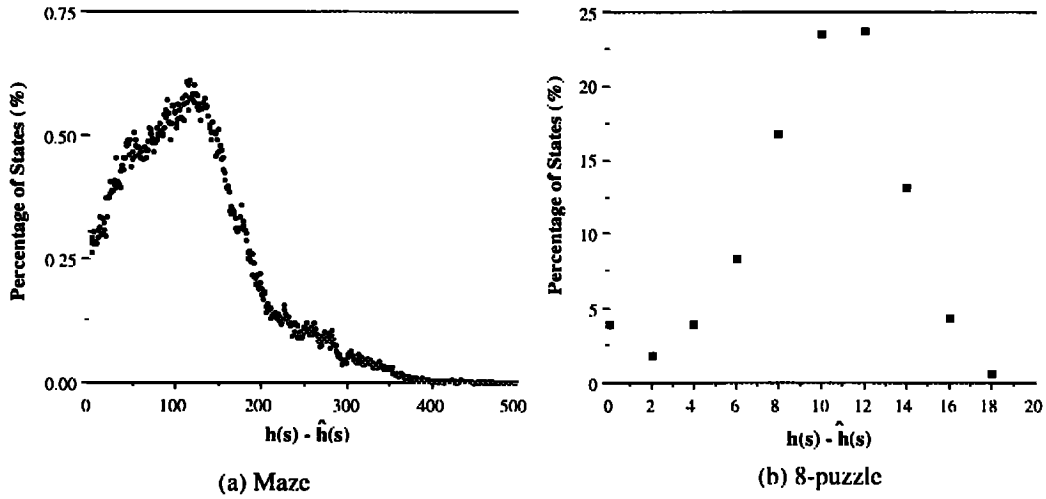


Figure 5: Distribution of heuristic depressions.

tic search and the organizational approach in multi-agent environment.

References

Banerji, R. B. 1980. *Artificial Intelligence: A Theoretical Approach*. New York: Elsevier North Holland.

Ishida, T. and Korf, R. E. 1991. Moving Target Search. In Proceedings of the Twelfth International Joint Conference on Artificial Intelligence, 204 210. International Joint Conferences on Artificial Intelligence, Inc.

Ishida, T. 1995. Two is not Always Better than One: Experience in Real-Time Bidirectional Search. In Proceedings of the First International Conference on Multi-Agent Systems, 185 192. American Association for Artificial Intelligence.

Knight, K. 1993. Are Many Reactive Agents Better Than a Few Deliberative Ones? In Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence, 432–437. International Joint Conferences on Artificial Intelligence, Inc.

Korf, R. E. 1985. Depth-First Iterative-Deepening: An Optimal Admissible Tree Search. *Artificial Intelligence* 27:97–109.

Korf, R. E. 1990. Real-Time Heuristic Search. *Artificial Intelligence* 42:189–211.

Nilsson, N. J. 1971. *Problem Solving Methods in Artificial Intelligence*. New York: McGraw-Hill.

Pearl, J. 1984. *Heuristics*. Reading, Mass.: Addison-Wesley.