Chin. Phys. B Vol. 20, No. 4 (2011) 048103

Memoryless cooperative graph search based on the simulated annealing algorithm

Hou Jian(候 健), Yan Gang-Feng(颜钢锋), and Fan Zhen(樊 臻)†

Department of Systems Science and Engineering, Zhejiang University, Hangzhou 310027, *China*

(Received 24 October 2010; revised manuscript received 16 November 2010)

We have studied the problem of reaching a globally optimal segment for a graph-like environment with a single or a group of autonomous mobile agents. Firstly, two efficient simulated-annealing-like algorithms are given for a single agent to solve the problem in a partially known environment and an unknown environment, respectively. It shows that under both proposed control strategies, the agent will eventually converge to a globally optimal segment with probability 1. Secondly, we use multi-agent searching to simultaneously reduce the computation complexity and accelerate convergence based on the algorithms we have given for a single agent. By exploiting graph partition, a gossip-consensus method based scheme is presented to update the key parameter—radius of the graph, ensuring that the agents spend much less time finding a globally optimal segment.

Keywords: search, simulated annealing, graph partition, globally optimal

PACS: 81.40.Ef, 05.65.+b, 93.85.Bc **DOI:** 10.1088/1674-1056/20/4/048103

1. Introduction

The search problems, one of the most common tasks in the world, dates back to World War 2 for finding targets or achieving missions. Most of the related research focuses on efficiency in terms of the amount of time spent searching or the total distance that the agent travels. In this paper we concentrate on how single or multiple memoryless agents search for a globally optimal segment in an uncertain environment, regardless of time or energy.

One motivating scenario is as follows: when mining for oil, gold or other rare metals, a team of agents are sent out to explore a complex, tough and uncertain environment. The agents need to find places that are likely to be filled with minerals and to relay a message with findings. Due to the abominable environment, it is difficult to construct a map of the search area. So we hope that the agents can search the region without the use of memories. Based on the probability choice of the next searching region, the agents gradually converge to a globally optimal segment (a place that has the greatest likelihood of being filled with minerals). In addition, the energy used to drive the agents can be gathered from the sun and is therefore not so important compared with depositing the limited mines. Hence, the goal of this paper is to exploit the probability-based memoryless agents to reach a globally optimal segment.

1.1. Historic overview and related literature

Searching for targets in an uncertain environment has been a topic of interest for several decades. Koopman,[1] Stone[2] were initially motivated by a desire to develop efficient search methods for finding enemy marine vessels. Until now, search strategies have been applied to both military and civilian applications, such as rescue operations in open sea or sparsely populated areas, searching for enemy targets, looking for mineral deposits, and so on.[3] The search problem is attracting more and more attention.

There has been some literature relating to the search problem in continuous time and space in which the initial target distributions are usually restricted to a very special set.[2,3] A more practical approach is to make a discretization of the search region and to substitute the region by a graph,[4] in which each vertex represents each part of the available searching regions and the edges denote the links between the regions. Then a segment describes a sequence of successive regions to search, along with the amount of resources or facilities that should be used in each region. So the problem becomes how to find the optimal vertices or segment in the graph. In recent years, a

†Corresponding author. E-mail: fanzhen0095019@gmail.com

© 2011 Chinese Physical Society and IOP Publishing Ltd

fair amount of work[5−9] has appeared to address these issues. An approximate solution based on the graph partition of the search region is studied in Ref. [5] to search for a stationary target. Riehl *et al.*[6] introduced a method of hierarchically decomposing a graph for single agent searches. Followed this work, Riehl *et al.*[7] in 2007 proposed a graph-based model predictive search algorithm using a similar approach to minimize the expected time until the target is detected. In addition to the graph partition strategy, the probability based strategy has also attracted some interest. In Ref. [8], Sujit and Ghose represented the prior knowledge of the location of the target as an uncertainty map, which could be a function of the probability distribution of the target's location on the graph vertices. Then a well-known k-shortest path algorithm used in network flow applications was proposed for UAVs searching for targets in an unknown environment and the results showed superiority over standard random search and greedy strategies. Chung *et al.*[9] achieved probabilistic modeling of a class of search and identification problems using Bayesian updating of probability maps, and solved these problem as a mixed-integer program.

As social development and science progresses, the problem of using a number of autonomous agents to cooperatively search for a target whose location is uncertain has gained increasing interest recently, but it is now even more complex. A short-horizon look-ahead approach is presented in Ref. [10] where the agents share all the information with each other. Riehl and Hespanha[11] introduced a method of hierarchical decomposition of a graph that finds close-to-optimal paths for a group of cooperative agents searching for one or more targets.

In addition to the general requirements of the search problem, such as the efficiency of the search algorithm usually measured by the time it takes to find the target and[12−14] the limitation of the search environment,[15,16] memory is also an important indicator, especially in tough search environments. Hence, memoryless or partially memorial agents provide another requirement for the system. Random walks on graph[17,18] is a common strategy for memoryless agent searching. However, it is time consuming and easily trapped into the local optimum. Hanusse *et al.* in Ref. [19] first provided an "ask and answer" problem in an unknown network (graph), where each node in this network will provide an advice by pointing at the first edge on a shortest path to the target. Unfortunately, some of the nodes, called liars, will give faulty information. Following this work, Hanusse *et al.* continued this idea. In Ref. [20], they proposed a randomized algorithm for the same problem, focusing mainly on its memory requirement, and gave the expected number of edges traversed by the agent before finding the target. As the search environment becomes harder and harder, people have become increasingly reliant on memory. Our paper continues this tough requirement.

1.2. Our work

Motivated by the scenario that a group of autonomous mobile agents are sent to search for a segment in an uncertain environment, the paper studies the problem of a single and a group of memoryless agents searching for a globally optimal segment. Each agent is assumed to be capable of computing the probability for finding the target in the regions (we denote it as the reward at the vertex) in its view. So the globally optimal segment means the maximal rewards sum of several successive vertices in the searching region. In addition, the searching region may be too tough to be constructed as a corresponding map, so memoryless agents are needed. In such a situation, the main problem becomes how to solve the traditional search problem in a discrete environment represented by a graph: how can an agent or a group of memoryless agents with limited sensing capabilities find the segment that maximizes the rewards subject to a cost constraint?

To solve the problem, we propose that each mobile agent searches for a segment that occupies several successive vertices based on an iterative algorithm, where each agent moves with a certain probability at each step. The probability that governs the movement is purely controlled by the rewards at the vertices in the graph. Through updating the positions of the agents, the probability of finding the target is going to become larger and larger. Eventually, at least one of the agents will converge to the vertex that is in the middle of a globally optimal segment in the area. In addition, we show that by proper graph partition, the time and computational complexity required to find the globally optimal segment can be greatly reduced.

1.3. Notations and organization

Throughout the paper, $\lceil x \rceil$ represents the minimal integer no less than x, and $\lfloor x \rfloor$ denotes the maximal integer that does not exceed x. The notation $[\cdot]^+$

means that it is zero when the value in the square bracket is less than zero and it has the same value otherwise.

The remainder of this paper is organized as follows. We formally formulate the search problem in Section 2. In Section 3, two efficient algorithms are given based on the simulated annealing in different environments separately. We consider the multi-agent cooperative search problem in Section 4. The simulation and the conclusion are given in Sections 5 and 6.

2. Graph search problem formulation

Given a connected and bounded searching region modeled by a graph $G := (V, E)$ with each vertex in V representing one part of the searching region and an edge $e \in E$ between two vertices meaning the corresponding two parts of region bourder each other. For any vertex $v \in V$, we denote the neighbour $v' \in N(v)$ if there is an edge $e \in E$ between vertices v and v'. Let $n(v)$ be the cardinality of the set $N(v)$.

We define k-way partition $\overline{V} := \{\overline{V}_1, \overline{V}_2, \ldots, \overline{V}_k\}$ of G as a set of disjoint subsets of V such that $\overline{V}_1 \bigcup \overline{V}_2 \bigcup \cdots \bigcup \overline{V}_k = V$. These subset vertices $\overline{V}_i$ $(i = 1, 2, \ldots, k)$ are known as meta-vertices.[11] A meta-graph $\overline{G} := (\overline{V}, \overline{E})$ induced by the partition $\overline{V}$ has an edge $\overline{e} \in \overline{E}$ between meta-vertices $\overline{V}_i, \overline{V}_j \in \overline{V}$ if and only if G has at least one edge $e \in E$ between $v \in \overline{V}_i$ and $v' \in \overline{V}_j$. Figure 1 shows an example of a meta-graph.

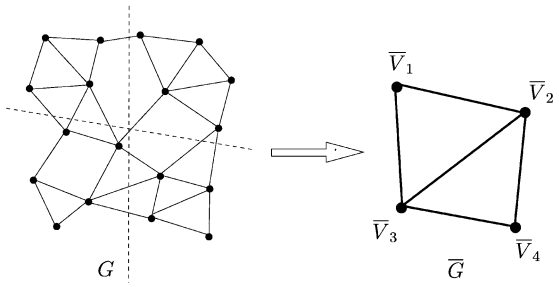


Fig. 1. An example of 4-way partition.

Consider one or a number of agents searching in the bounded region for a segment of consecutive regions, which is represented by m successive vertices in the graph $G := (V, E)$. Each vertex v_i owns a non-negative reward $r(v_i)$ (e.g., the probability of finding the target). Our goal is to find an m steps segment

$$p := (v_1, v_2, \ldots, v_{m-1}, v_m) \quad (v_i, v_{i+1}) \in E,$$
$$\forall\, i = 1, 2, \ldots, m-1$$

that maximizes the total segment rewards

$$F_p = \sum_{i=1}^{m} r(v_i). \tag{1}$$

We assume that each agent i has a disc sensing radius R_i. Sensing radius R_i and R_j may not be required to be the same for $i \neq j$. Among all the agents, we denote $R_{\min}$ as the minimal R_i, i.e., $R_{\min} = \min_i(R_i)$. Let d be the maximal distance between two adjacent vertices. We discuss in this paper under the assumption that the sensing region is large enough, i.e., $2R_{\min} \geq (m+1)d$. For each vertex $v \in V$, we provide a new value $f(v)$ that is the rewards sum of the maximal m steps segment with vertex v in the middle of the segment, that is,

$$f(v) = \max_v F_p$$

subject to an m successive vertices segment $p := (v_1, v_2, \ldots, v_{m-1}, v_m)$ where $v = v_{m/2}$ when m is even, or $v = v_{(m+1)/2}$ when m is odd.

3. Single agent search with a large enough sensing region

In this section, we will discuss a single memoryless agent searching for a segment in both a partially known environment and an unknown environment, respectively. We have assumed that the above sensing region is large enough such that $2R_{\min} \geq (m+1)d$. This means that when an agent is at vertex v, it can obtain not only its own value $f(v)$, but also the values $f(v')$ of its neighbours, where $v' \in N(v)$.

3.1. In a partially known environment

Let V_m be the set of vertices that are local maxima, i.e.,

$$V_m \triangleq \{v \in V | f(v') < f(v), \quad \forall v' \in N(v)\},$$

then the radius of the graph G is

$$r_G \triangleq \min_{v \in (V/V_m)} \max_{v' \in V} d(v, v'),$$

where $d(v, v')$ is the minimal number of edges from v to v', and $v \in V/V_m$ means $v \in V$ but $v \notin V_m$ in which $V_m \subset V$. In this subsection, we assume that the environment is partially known such that the graph radius r_G could be estimated.

Now we first give the algorithm for one agent searching for an optimal segment based on the simulated annealing algorithm.[21−23]

Algorithm 1

Step 1 Randomly initialize the location of the agent and set time $t = t_0$.

Step 2 Suppose the agent can obtain the value $f(v)$ at current location v and the ones $f(v')$ of its neighbours, $v' \in N(v)$, then it moves to one of the new locations $v' \in N(v)$ according to the following probability

$$q_{v\to v'} = \frac{1}{n(v)} \exp \frac{-[f(v) - f(v')]^+}{\alpha(t)}. \tag{2}$$

Otherwise, it stays stationary.

Step 3 $t = t + 1$. Go to Step 2.

END

In Algorithm 1, $\alpha(t)$ is a positive parameter depending on time t, that we are going to design later. Thus, the probability for the agent to remain in the current location v is

$$1 - \sum_{v' \in N(v)} q_{v\to v'}.$$

This updated rule means that if the agent moves to an adjacent location for which the segment has a worse value, then the probability of going to that location decreases in terms of an exponential function.

Then the stochastic process generated by Algorithm 1 is a time-inhomogeneous Markov chain whose transition probability matrix at time t is given by

$$P_{vv'}(t) = \begin{cases} 0, & \text{if } v' \notin N(v) \cup \{v\}, \\ \dfrac{1}{n(v)} \exp \dfrac{-[f(v) - f(v')]^+}{\alpha(t)}, & \text{if } v' \in N(v), \\ 1 - \sum\limits_{u \neq v} P_{vu}(t), & \text{if } v' = v. \end{cases} \tag{3}$$

Now we present the following theorem showing that the agent converges to a globally optimal segment maximizing the total rewards when the updated rule we proposed is applied with the proof referred to in the simulated annealing algorithm.[22,24]

Theorem 1 Suppose that a memoryless agent is initially distributed in a given graph G with the radius of the graph r_G. If $\alpha(t) > 0$ is non-increasing and satisfies

$$\alpha(0) > \alpha(1) > \cdots > \lim_{t\to\infty} \alpha(t) = 0, \tag{4a}$$

$$\alpha(t) \geq \frac{r_G \cdot D}{\ln(t+1)}, \tag{4b}$$

in which D is defined as

$$D = \max_{v' \in N(v)} |f(v) - f(v')|,$$

then the agent converges to a globally optimal vertex (in the middle of a globally optimal segment) maximizing the rewards subject to the m steps constraint with probability 1.

Remark 1 The monotonicity of $\alpha(t)$ decreasing to zero ensures that the agent gets increasingly few opportunities to enter a bad location, while condition (4b) guarantees convergence to a globally optimal vertex rather than being trapped in a locally optimal one.

3.2. In an unknown environment

In Algorithm 1, we consider the case when the bounded region substituted by a graph is partially known, which means that we can obtain the corresponding radius of the graph. In this subsection, we focus on a case where the searching region is completely unknown. Instead, the agent must mark the number of times it visits for each vertex. Also, we require that the sensing region is large enough such that we can obtain the value f at its current location and the ones of its neighbours.

Note that after K iterations, X_K is the current state of the Markov chain generated by the algorithm and $V_K(v)$ is the number of times the Markov chain $\{X_K\}$ has visited state v after K iterations. For all $v \in V$, $X_K^\star$ is the state that maximizes $V_K(v)/n(v)$ and $V^\star$ is the globally optimal value vertices set. Now we state a modification of the simulated annealing algorithm for solving the search problem in an unknown environment.

Algorithm 2

Step 1 Given the initial location $X_0 \in V$, let $V_0(X_0) = 1$ and $V_0(v) = 0$ for all $v \in V/\{X_0\}$. Let

$K = 0$ and $X_K^\star = X_0$.

Step 2 Suppose the agent can obtain the value $f(X_0)$ at current location X_0 and the ones $f(v')$ of its neighbours, $v' \in N(X_0)$, then it moves to one of the new locations $v' \in N(X_0)$ according to the following probability

$$q_{X_0 \to v'} = \frac{1}{n(X_0)} \exp \frac{-[f(v) - f(v')]^+}{\alpha}, \qquad (5)$$

in which α is a positive constant.

Step 3 Let $K = K+1$, $V_K(X_K) = V_{K-1}(X_K)+1$, and $V_K(v) = V_{K-1}(v)$ for all $v \in V/\{X_K\}$. If $V_K(X_K)/D(X_K) > V_K(X_{K-1}^\star)/D(X_{K-1}^\star)$, then let $X_K^\star = X_K$; otherwise, let $X_K^\star = X_{K-1}^\star$. Go to Step 2.

END

The stochastic process $\{X_K\}$ generated by Algorithm 2 is a time-inhomogeneous Markov chain whose transition probability matrices are given by

$$P_{vv'}(K) = P\left[X_{K+1} = v' | X_K = v\right] = \begin{cases} 0, & \text{if } v' \notin N(v) \cup \{v\}, \\ \dfrac{1}{n(v)} \exp \dfrac{-[f(v) - f(v')]^+}{\alpha}, & \text{if } v' \in N(v), \\ 1 - \sum\limits_{u \neq v} P_{vu}(t), & \text{if } v' = v. \end{cases} \qquad (6)$$

Now we have the following convergence theorem with the proof referred to Ref. [25].

Theorem 2 Suppose that a partially memorial agent is initially randomly distributed in an unknown region and it updates its location under Algorithm 2. Then the sequence $\{X_K^\star\}$ converges almost certainly to the set $V^\star$ (in the sense that there is a set A such that $P(A) = 1$, and for all $\omega \in A$, there is $K_\omega > 0$ such that $X_K^\star(\omega) \in V^\star$ for all $K \geq K_\omega$).

Remark 2 It is easily obtained from Algorithm 2 that the agent is more likely to visit parts of the vertices, but in the risk of being trapped into the local optima when α is small. When α is large, the agent is able to visit any part of the graph, but it may need too much time to focus on the necessary globally optimal segment. So, as a single agent searches in unknown environment based on Algorithm 2, how to choose a proper α is an important issue.

4. Multi-agent search with large enough sensing region

In this section, we study the conditions for a team of $k(> 1)$ agents to accelerate searching for an optimal segment in a partially known graph. As we have obtained from Theorem 1, the key parameter $\alpha(t)$ is related to the radius of the graph, so the computation time for Theorem 1 is extraordinary long if the searching region is large, especially in the case of a large graph radius. Then a cooperative multi-agent search is required in order to solve the problem.

Our strategy is to make a k-way partition $\overline{V} := \{\overline{V}_1, \overline{V}_2, \ldots, \overline{V}_k\}$ of graph G as a set of disjointed subsets of V such that each agent i searches the vertices in its own subset $\overline{V}_i$. Then the time-varying parameter $\alpha(t)$ is only related to the sub-radius $r_{\overline{G}_i}$ of the sub-graph $\overline{G}_i := (\overline{V}_i, \overline{E}_i)$. Our problem now is how to partition the graph to get the minimal sub-radius, i.e.,

$$r_{\text{best}} = \min_{\text{partition}} \max_{\overline{G}_i} r_{\overline{G}_i}.$$

Now we propose a distributed averaging algorithm based on the gossip strategy to solve the partition problem for a given graph.

Algorithm 3

Step 1 Initially, we randomly give a k-way partition $\overline{V}$. Compute each part's sub-radius $r_{\overline{G}_i}, i = 1, \ldots, k$, time $t = 0$;

Step 2 At time t, we randomly select two adjacent sub-graphs $\overline{G}_i$ and $\overline{G}_j$. Let $S_{ij}(t) = r_{\overline{G}_i}(t) + r_{\overline{G}_j}(t)$ and $D_{ij}(t) = |r_{\overline{G}_i}(t) - r_{\overline{G}_j}(t)|$. If any vertices in one of these two sub-graphs are repartitioned into the other sub-graph such that $S_{ij}(t)$ decreases, then repartition these two sub-graphs to decrease this value. Otherwise, this value is left invariable;

Step 3 If $D_{ij}(t) = 0$, go to Step 4. If $D_{ij}(t) > 1$, we repartition such that $D_{ij}(t+1) < D_{ij}(t)$, and if $D_{ij}(t) = 1$, we repartition such that $r_{\overline{G}_i}(t+1) = r_{\overline{G}_j}(t)$ and $r_{\overline{G}_j}(t+1) = r_{\overline{G}_i}(t)$. This update is called a swap;

Step 4 $t = t + 1$, go to Step 2.

END

Now we present our graph partition theorem to minimize the sub-radius.

Theorem 3 Given a graph $G := (V, E)$ with radius r_G, we partition this graph into k disjoint subsets under Algorithm 3. If the graph G is quasi-balanced, i.e., each iteration at Step 3 can be achieved when $D_{ij}(t) \geq 1$, then each subset contains roughly the same sub-radius. Moreover, we obtain the minimal sub-radius r_{best}.

In this paper, we let $m(t) = \min_i r_{\overline{G}_i}(t)$, $M(t) = \max_i r_{\overline{G}_i}(t)$ and $D(t) = M(t) - m(t)$. We need the following two lemmas to prove our theorem.

Lemma 1[26] Consider a distributed algorithm in which, for any graph G with k vertices, in addition to the constraint that the value x_i, $i = 1, \ldots, k$ at each vertex is always an integer and the sum of the k values in the network does not change with time. The following conditions are met.

(C1) For any given initial condition $x(0) = [x_1(0), \ldots, x_k(0)]$, at any time during the execution of the algorithm, the value of $x(t)$ lies in some finite set ψ (which may depend on $x(0)$);

(C2) For any state $x(t) = x$, there exists a finite time t_x such that $P[x(t+t_x) \in \varphi | x(t) = x] > 0$, where $\varphi = \{x | x_i \in \{L, L+1\}, i = 1, 2, \ldots, k, \sum_{i=1}^{k} x_i = S\}$, in which S can be written as $kL + R$, and L, R are integers with $0 \leq R < k$;

(C3) If $x(t) \in \varphi$, then $x(t') \in \varphi$ for all $t' > t$.

Such an algorithm converges to quantized consensus for any graph G and any initial condition $x(0)$.

Lemma 2[26] When there is quantized consensus, at any time t two vertices are selected. If the value difference of the two vertices are bigger than 1, then shorten the difference; if the difference equals 1, then do swap; or else do nothing. Let $N_{\max}(t) = |i | x_i(t) = M(t)|$ be the number of vertices with the maximum value in the network at time t. Then, for any time t such that $D(t) \geq 2$,

1) if $N_{\max}(t) > 1$, there always exists some time $t' \geq t$ such that there is a positive probability that $N_{\max}(t') < N_{\max}(t)$, and

2) if $N_{\max}(t) = 1$, then there always exists some time $t' > t$ such that there is a positive probability that $M(t') < M(t)$ and therefore $D(t') < D(t)$.

Proof of Theorem 3 Since $S_{ij}(t)$ is non-increasing and bounded, we can define time t_0 such that when $t \geq t_0$, $S_{ij}(t)$ does not change anymore. It is obvious from our algorithm that $m(t)$ is non-decreasing, $M(t)$ is non-increasing and so $D(t)$ is non-increasing. Therefore, at any time $t \geq t_0$, $r_{\overline{G}_i}(t)$ lies in some finite set ψ with the value between $m(t_0)$ and $M(t_0)$. So (C1) is satisfied. Also, if $r_{\overline{G}_i}(t) \in \varphi$ then $r_{\overline{G}_i}(t') \in \varphi$ for any $t' \geq t \geq t_0$ such that (C3) is met.

From Lemma 2, we construct a sequence of times $t_0, t_1, \ldots$ as follows. For each $i \geq 0$, if $N_{\max}(t_i) > 1$, then we let t_{i+1} be the first time for which there is a positive probability that $N_{\max}(t_{i+1}) < N_{\max}(t_i)$. Then there exists an integer l that $N_{\max}(t_l) = 1$, then $D(t_{l+1}) < D(t_l)$. So for any time t such that $D(t) \geq 2$, there is a finite time t' such that there exists a positive possibility that $D(t + t') \leq 1$, that is, $x(t + t') \in \varphi$, all the k sub-sets contain roughly the same sub-radius $r_{\overline{G}_i}$, $i = 1, \ldots, k$. This proves the condition (C2) of Lemma 1. Thus the gossip algorithm converges to quantized consensus, which means each subset contains roughly the same sub-radius. In addition, it is obtained that $S_{ij}(t)$ cannot decrease anymore, so the radius we obtain is the minimal sub-graph sub-radius r_{best}. The proof is completed.

Remark 3 We have assumed that the radius of the given graph G is r_G, which means the maximal distance of two vertices $d_{\max}$ is $2r_G$ or $2r_G - 1$. When we divide the graph into k sub-graphs, then there must be at least a distance of $\lceil d_{\max}/k \rceil$ in one of the sub-graphs, so the upper bound of the minimal sub-radius with k decomposing is $\widetilde{r}_{\text{best}} = \lfloor \lceil d_{\max}/k \rceil / 2 \rfloor$.

Romeo *et al.*[27] in their paper have shown that after n iterations, the distance of the state probability distribution from the optimal value $e_\star$ under the simulated annealing algorithm is given by

$$\|p(n) - e_\star\| = O\left(\left(\frac{1}{n}\right)^{\min(a,b)}\right), \tag{7}$$

where a and b are both related to radius r_G and the larger the $1/r_G$ is, the much larger both a and b are. So almost always is the efficiency that k agents in the sub-graphs search for an optimal segment, much higher than k agents searching alone in the whole searching region.

5. Simulations

In this section, we present several simulation results to validate Algorithm 3. For simplicity in analysis, we always define $f(v) = f(v')$ for any two vertices $v \neq v'$, which means each vertex owns the same reward.

We first provide a simulation to illustrate that our algorithm can partition the graph as evenly as possible when the graph is quasi-balanced. An initial partition of a graph with sub-radius 4, 10, 9 and 9 is given in Fig. 2. Applying the algorithm, then it is obtained that all the sub-radius converge to 8 (see Fig. 3).

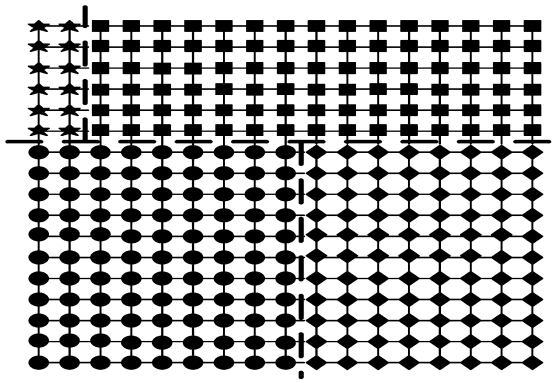


Fig. 2. Initial partition of a quasi-balanced graph.

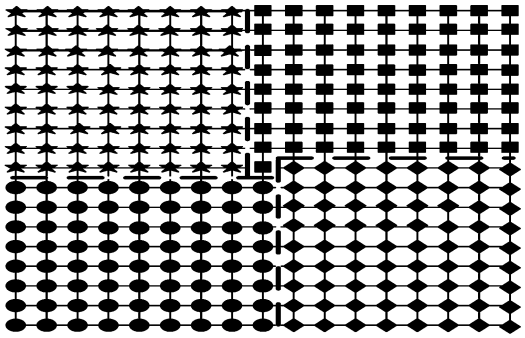


Fig. 3. The final partition of the given graph after applying Algorithm 3.

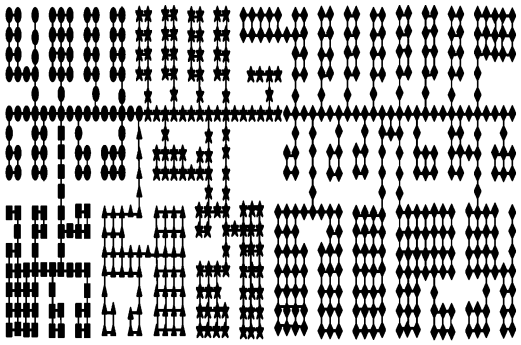


Fig. 4. Initial partition of the third floor into 5 subgraphs.

To verify our algorithm in a regular graph, we consider a model of the third floor of Harold Frank Hall at UCSB, where the floor has been divided into 646 cells, each about 4 square meters in size. The initial partition and the final partition of the floor are shown in Fig. 4 and Fig. 5, respectively. From the simulation results, we observe that our partition strategy also works well in a regular graph.

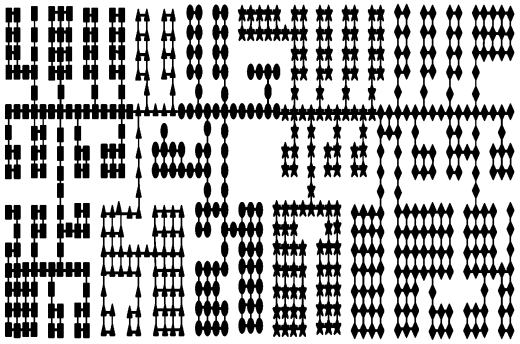


Fig. 5. The final partition of the third floor after applying Algorithm 3.

6. Conclusion

This paper utilizes a simulated annealing algorithm to solve a traditional search problem in a discrete environment substituted by a graph. We focus on a complex and tough environment where the agents are not capable of creating a map of the search region or able to remember all of the paths that they have passed. In each step, each agent searches in its view and changes its location to one of its adjacent positions with probability q, which only depends on the rewards of the vertices and the radius of the searching graph. We utilize the graph partition method and the gossip algorithm to optimize the sub-radius to speed up convergence to a globally optimal segment (at least one agent in the middle of the segment), at the same time avoiding obstacles and threats.

We are currently investigating several related issues for a future work. An important question that remains unanswered is what to do if the radius of the searching graph is uncertain. Another interesting problem is how to search for multiple optimal segments.

References

[1] Koopman B O 1946 *Operations Evaluations Group Report*

[2] Stone L D 1975 *Theory of Optimal Search* (New York: Academic Press) p. 1

[3] Ieong S, Lambert N, Shoham Y and Brafman R 2007

Proceedings of the National Conference on Artificial Intelligence, British Columbia, Canada, July 22–26, 2007 p. 1158

[4] Liu F and Xu X P 2007 *Chin. Phys.* **16** 282

[5] DasGupta B, Hespanha J P, Riehl J R and Sontag E 2006 *Nonlinear Anal. Theor.* E **65** 1773

[6] Riehl J R and Hespanha J P 2005 *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference*, Seville, Spain, December 12–15, 2005 p. 2188

[7] Riehl J R, Collins G E and Hespanha J P 2007 *Proceedings of the 2007 IEEE Conference on Decision and Control*, New Orleans, USA, December 12–14, 2007 p. 2998

[8] Sujit P B and Ghose D 2004 *IEEE. T. Aero. Elec. Sys.* E **40** 491

[9] Chung T H, Kress M and Royset J O 2009 *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, Kobe, Japan, May 12–17, 2009 p. 243

[10] Polycarpou M M, Yang Y and Passino K M 2001 *Proceedings of the 2001 IEEE International Symposium on Intelligent Control*, Mexico City, Mexico, September 5–7, 2001 p. 1

[11] Riehl J R and Hespanha J P 2007 *Proceedings of the 2007 American Control Conference*, New York, USA, July 9–13, 2007 p. 2557

[12] Avetisyan V V 2002 *J. Comput. Sys. Sci. Int.* E **41** 57

[13] Blum T and Kohlbacher O 2008 *Bioinformatics* E **24** 2108

[14] Vlachos M, Kozat S S and Yu P S 2010 *Acm. T. Web.* E **4** 1

[15] Guruprasad K R and Ghose D 2007 *Proceedings of the 1st International Conference on Advances in Control and Optimization of Dynamical Systems*, Bangalore, India, February 1–2, 2007 p. 380

[16] Sujit P B and Ghose D 2004 *Proceedings of the 2004 American Control Conference*, Boston, USA, June 30–July 2, 2004 p. 5564

[17] Göbel F and Jagers A A 1974 *Stoch. Proc. Appl.* E **2** 311

[18] Lovász L 1993 *Combinatorics* E **2** 1

[19] Hanusse N, Kranakis E and Krizanc D 2004 *Discrete Appl. Math.* E **137** 69

[20] Hanusse N, Kavvadias D, Kranakis E and Krizanc D 2008 *Theor. Comput. Sci.* E **402** 190

[21] Zhao Z J, Zheng S L, Xu C Y and Kong X Z 2007 *Chin. Phys.* **16** 1619

[22] Kirkpatrick S 1984 *J. Stat. Phys.* E **34** 975

[23] Wang L and Smith K 1998 *IEEE T. Neural Network* E **9** 716

[24] Aarts E H L and Laarhoven P J M 2008 *Stat. Neerl.* E **43** 31

[25] Alrefaei M H and Andradóttir S 1999 *Manage. Sci.* E **45** 748

[26] Kashyap A, Basar T and Srikant R 2007 *Algorithmica* E **43** 1192

[27] Romeo F and Sangiovanni-Vincentelli A 1991 *Algorithmica* E **6** 302