International Journal of Innovative
Computing, Information and Control
Volume **10**, Number **6**, December **2014**

ICIC International ©2014 ISSN 1349-4198
pp. 2237–2249

A DYNAMIC PLANNING METHOD OF MOBILE AGENT PATH BASED ON WINDOW STRATEGY

Xiaolin Wang*, Guangzhou Zeng and Xinshun Xu

School of Computer Science and Technology
Shandong University
No. 1500, Shunhua Road, Jinan 250101, P. R. China
*Corresponding author: xlwang@sdu.edu.cn; { gzzeng; xuxinshun }@sdu.edu.cn

Received January 2014; revised May 2014

Abstract. *The mobile agent is a user program that is able to migrate continuously among sites in a network and use site services to complete task. In a social cooperative system, the site can be regarded as a proxy of the social member and all sites form a work environment for mobile agent. The sequence of sites on which the mobile agent completed tasks is called mobile agent path. The path planning has been one of the key issues in the mobile agent research. This paper discusses a mobile agent path planning model based on Markov decision process, and proposes a dynamic mobile agent path planning method based on the referral network with window strategy, of which the referral network is used to characterize the partial work environment of mobile agent, the window strategy is used to find the optimal workplace on the referral network, similar to a 'thinking for multi-step before taking one step' behavior mode in the chess game. The simulation results and an example have shown that, by utilizing the referral network with window strategy, the mobile agent is able to dynamically construct an optimal path for executing sequence of tasks.*

Keywords: Mobile agent, Path planning, Markov decision process, Referral network, Window strategy

1. Introduction.

The mobile agent is a user program that is able to migrate continuously among sites in a network and use local services to execute task, such as information retrieval, workflow management, intelligent robots and wireless sensor network. In a mobile agent system, the site where the mobile agent performs its task is called the workplace, and the sequence of workplaces is called the mobile agent path. Since mobile agent migration must be good for global goal resolving, the optimal site choice, i.e., path planning, has been one of the key issues in the mobile agent research.

The approaches to mobile agent path planning can be divided into static planning methods and dynamic planning methods [1,2]. The static planning refers to methods in which the path is generated before the mobile agent is sent to migrate. By interpreting the path knowledge carried with it, the mobile agent is able to know where it should go to execute task. The dynamic planning refers to methods in which the mobile agent has no or very little known path knowledge before it starts migrating, and it has to dynamically choose the workplace according to its goal and the environment exploration.

The most commonly used static planning methods are itinerary method [3-5] and Traveling Agent Problem method [6-8]. Since the static planning method only uses the known environment information before the mobile agent starts migrating, the path cannot adapt well to the environment changes, such as network disconnection, host failure, and services change.

In dynamic planning methods, the service workplace can either be discovered by mobile agent itself [9-13], or obtained through site recommendation [14-16]. Since the workplace discovery method requires sufficient knowledge and capability for environment exploration, it leads to a larger size of the mobile agent. The larger the mobile agent is, the longer it takes to migrate, and the higher probability that migration failures. The workplace recommendation method means that the site explores environment and recommends the workplace to the mobile agent, therefore, the mobile agent can be light weighted since it does not need to carry the knowledge and codes for environment exploration.

In practice, not only the workplace discovery methods but also the workplace recommendation methods, are almost adopted the one step strategy, i.e., the environment exploration only takes care of the current task to be executed and does not consider any influence of remained tasks on the workplace choice, which is similar to a 'thinking for one step before taking one step' behavior in the chess game. Therefore, the one step strategy lacks the ability for whole path optimization.

In a social cooperative application, the site can be considered as the proxy of a social member and all sites form a work environment for mobile agent. Assumed that each site can only service one task, then for a given sequence of tasks, the mobile agent migration process is a sequence of discrete time events and can be modeled by Markov decision process (MDP). This paper discusses a mobile agent path planning model based on MDP, and proposes a dynamic mobile agent path planning method based on the referral network [17,18] with window strategy, of which the referral network is used to characterize the partial work environment of mobile agent, the window strategy is used to find the optimal workplace on the referral network, similar to a 'thinking for multi-step before taking one step' behavior mode in the chess game. Since the window strategy considers multiple task contributions to optimal workplace choice on the referral network and a number of referral networks form a global environment, it is a partial-global optimization method in essence and can make a better path for mobile agent than one step strategy.

The structure of the following parts of this paper is: Section 2 defines the path of a mobile agent, and discusses the mobile agent path planning model based on MDP; Section 3 gives the mobile agent path planning algorithm based on the referral network model with window strategy; Section 4 discusses the simulation results of the algorithm given in Section 3, and gives an example; Section 5 compares related work, followed by a summary and a prospect for the future in Section 6.

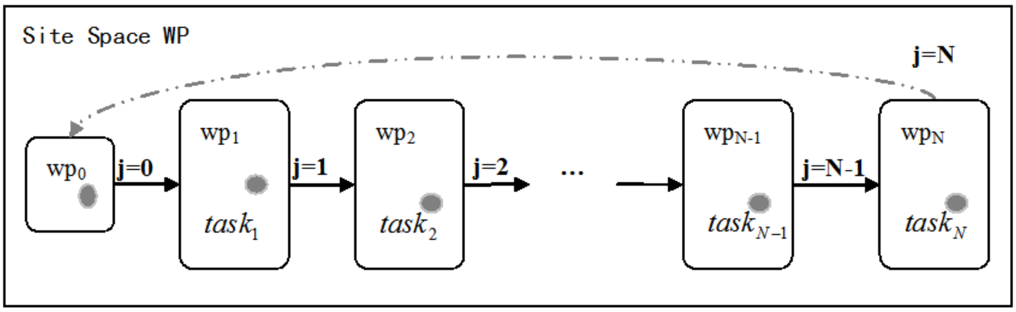
2. MDP-based Mobile Agent Path Planning Model.

2.1. Definition of mobile agent path and MDP model.

Let the goal of mobile agent be completing a sequential business process, denoted by $SBP = (task_1,\ task_2,\ \ldots,\ task_N)$ with $N \geq 1$, of which $task_1$ is the beginning task, and $task_N$ is the ending task. Let WP be the site space, of which each site is a proxy of the social member and can only serve one task of SBP every time. In order to make distinction, the sites where the mobile agent launched or the mobile agent completed task are named the workplace.

Definition 2.1. *Given SBP, the mobile agent path is a sequence of workplaces with no task failure, denoted by* $path = (wp_0, wp_1, wp_2, \ldots, wp_N)$*, of which* wp_0 *is the site where the mobile agent launched as well as returned, and the mobile agent did not execute any tasks; except for* wp_0*,* wp_j *is the unique workplace where* $task_j$ *has been completed.*

Figure 1 gives an illustration of Definition 2.1, where WP refers to the site space, and $j = 0, 1, 2, \ldots, N$ labels the order that the mobile agent leaves wp_0, wp_1, wp_2, $\ldots$, wp_N respectively. For the convenience of discussion, the order j is named the migrating-out



Note that since the mobile agent returning is a default act, the movement from WP_N to WP_0 is not considered as part of the path.

FIGURE 1. Graphic explanation of mobile agent path

time and called time j for short, wp_j is named the current workplace of mobile agent at time j.

Property 2.1. *Except wp_0 and wp_N, all the other workplaces on the mobile agent path play two roles, of which the former is a service provider while the mobile agent moved on it, and the later is a workplace recommender when the mobile agent intends to leave it.*

It is obvious from Figure 1 that, wp_0 is the launched place of the mobile agent, it needs not provide service to task, and wp_N is the last workplace of the mobile agent, it needs not recommend new workplace to mobile agent. Except wp_0 and wp_N, wp_j provides services to $task_j$ at first, and then recommends the workplace for $task_{j+1}$ to mobile agent, the role has changed at time j.

At time j, let the tasks remained in *SBP* be the state, denoted by $s_j = (task_{j+1}, task_{j+2}, \ldots, task_N)$.

Property 2.2. *The state s_j is only related to time j, and has no relationship with the migrating history of mobile agent prior to time j.*

According to Definition 2.1, since there is no task failure on the mobile agent path, $s_0 = (task_1, task_2, \ldots, task_N)$, $s_1 = (task_2, \ldots, task_N), \ldots, s_{N-1} = (task_N)$, therefore, s_j is only related to time j, and has no relationship with migrating history of mobile agent prior to time j.

Let a_i be an act that the current workplace wp_j of mobile agent recommends a workplace wp for executing $task_{j+1}$ under state s_j, denoted by $a_i = (wp, task_{j+1}|wp_j, s_j)$, where $task_{j+1}$ is the first task of s_j.

Definition 2.2. *Given $task_{j+1} \in s_j$ at time j, the workplace recommendation process of wp_j for $task_{j+1}$ can be described as a sequence of acts, denoted by $A = (a_{1,} a_2, \ldots, a_M)$.*

Property 2.3. *For all $i = 1, 2, \ldots, M$, $a_i = (wp, task_{j+1}|wp_j, s_j)$ is random.*

Based on the definition of $a_i = (wp, task_{j+1}|wp_j, s_j)$, the randomness of a_i is equivalent to the randomness of *wp*. Since there is service uncertainty with each site in the *WP* space, it is random whether the site can be recommended as *wp* by wp_j for $task_{j+1}$. Therefore, a_i is random.

According to Definition 2.1, since the mobile agent path does not include any sites where the mobile agent failed to execute task, the completion of $task_{j+1}$ on *wp* will definitely move *SBP* from state s_j to state s_{j+1}. As a result of the randomness of *wp* being recommended by wp_j, the state transition of *SBP* based on *wp* is also random.

According to Property 2.2 and Property 2.3, the process of mobile agent path planning could be described with MDP model.

Definition 2.3. *Given $SBP = (task_1, task_2, \ldots, task_N)$, the problem of mobile agent path planning can be described with a 4-tuple (S, A, P, C), where*

$S = \{s_0, s_1, s_2, \ldots, s_{N-1}\}$ *is the set of states,* $\forall s_j \in S$, $s_j = (task_{j+1}, task_{j+2}, \ldots, task_N)$ *is a task sequence remained in SBP at time j;*

$A = \{a_1, a_2, \ldots, a_M\}$ *is the set of acts,* $\forall a_i \in A$, $a_i = (wp, task_{j+1}|wp_j, s_j)$ *shows that wp_j recommends wp as the execution place of $task_{j+1}$ under state s_j at time j;*

$P : S \times A \times S \to [0, 1]$ *is the state transition function, of which $p(s_{j+1}|s_j, a)$ is the state transition probability from s_j to s_{j+1} by taking act a at time j, which can be equivalently expressed by $p(wp, task_{j+1}|wp_j, s_j)$, i.e., the probability of wp_j recommending wp as the execution place of $task_{j+1}$ at time j;*

$C : S \times A \times S \to C$ *is the payment function, of which $c(s_{j+1}|s_j, a)$ is the instant cost to make state transition from s_j to s_{j+1} by taking act a at time j, which can be equivalently expressed by $c(wp, task_{j+1}|wp_j, s_j)$, i.e., the cost of $task_{j+1}$ on wp recommended by wp_j at time j.*

According to Definition 2.1 and Definition 2.3, the MDP-based process of mobile agent path planning can be explained with Figure 2.

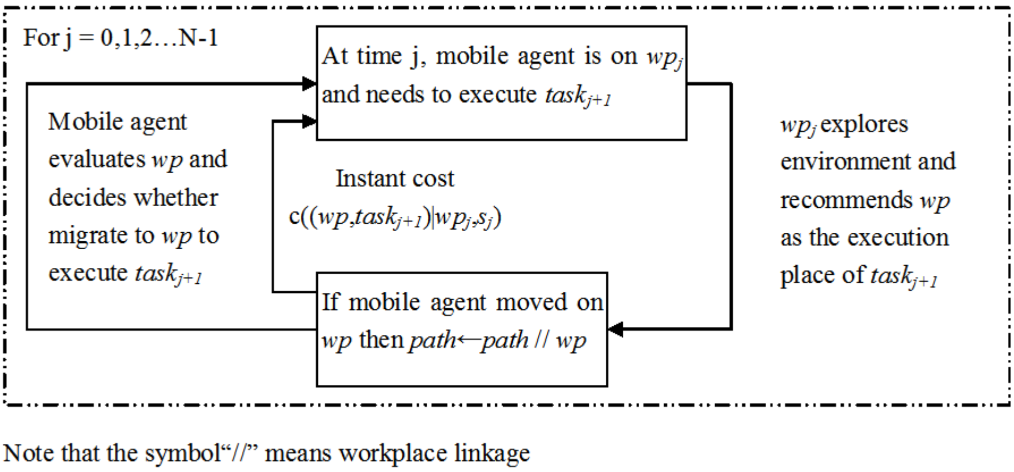


Figure 2. MDP-based mobile agent path planning model

In Figure 2, the environment explored by wp_j can be global *WP* or a sub-space of *WP*. In order to keep the global environment being open, this paper assumes that the environment explored by wp_j is a sub-space of *WP* and named the sub-space of *WP* as an acquaintance network.

Definition 2.4. *At each time j, the acquaintance network of wp_j is defined as $Knet_j = (V, E)$, where $V \subseteq WP \wedge wp_j \in V$ is the set of sites and each site is a proxy of the social member; E is the set of edges, $\forall e \in E$, $e = (wp_p, wp_q)$ represents the cooperative relation between wp_p and wp_q.*

Definition 2.5. *The maximum path length between wp_j and the other entire site on $Knet_j$ is called the network diameter of $Knet_j$, denoted by D_j.*

Property 2.4. *$Knet_j$ can be constructed as a referral network.*

Following the results in social network research, $Knet_j$ is a referral network and can be constructed by Small World Model with 'Six Degree of Separation' principle [19,20]. If there is no site satisfying $task_{j+1}$, then it needs to enlarge D_j by introducing the indirect acquaintance relationships until the problem has been resolved.

According to Definition 2.3 and Figure 2, the total cost on the mobile agent path can be denoted by

$$C_{path} = \sum_{j=0}^{N-1} c((wp_{j+1}, task_{j+1})|wp_j, s_j) \tag{1}$$

Let the goal of mobile agent path planning be finding a minimal cost path through recommending the cost-minimized workplace by wp_j on $Knet_j$ at each time j, denoted by

$$C_{path*} = \min_{path \text{ on } WP}\{C_{path}\} \tag{2}$$

2.2. **Recommendation probability and instant cost of workplace.** $\forall task \in SBP$, let *tF* be the function requirement of *task*, of which *tF.inD* represents the input of *task* and *tF.outD* represents the output of *task*, with *tF.inD* and *tF.outD* both being disjunctive expressions of the parameters.

$\forall wp \in WP$, let (*sF*, *sC*) be a service of *wp*, of which *sF* is the service function and *sC* is the service cost. Let *sF.inD* be the input of *sF* and *sF.outD* be the output of *sF*, with *sF.inD* and *sF.outD* both being disjunctive expressions of the parameters.

Suppose that all concepts about *task* and *wp* are described by a uniform domain ontology, i.e., the concept with same name has the identical semantic explanation.

Definition 2.6. *If x_1 and x_2 are concepts with the same name then x_1 matches with x_2, denoted by match (x_1, x_2).*

For instance, *match*(*task.tF*, *wp.sF*) means that *task.tF* and *wp.sF* have the same function, and *match*(*tF.inD*, *sF.inD*) shows that *tF.inD* and *sF.inD* have one-to-one mapped parameters.

Let $tF.inD = (P^t_1 \wedge P^t_2 \wedge \ldots \wedge P^t_m)$ and $sF.inD = (P^s_1 \wedge P^s_2 \wedge \ldots \wedge P^s_m)$, of which the value of parameter P is $P.v$.

Definition 2.7. *If $match(tF.inD, sF.inD)$ and $\forall i, P^t_i.v = P^s_i.v$, it is defined that tF.inD equals to sF.inD, denoted by $eql(tF.inD, sF.inD)$.*

Similarly, there is *eql*(*tF.outD*, *sF.outD*).

Definition 2.8. *If $match(task.tF, wp.sF) \wedge eql(tF.inD, sF.inD) \Rightarrow eql(tF.outD, sF.out D)$ then wp is called an available workplace of task.*

Let *task* be the task to be executed at time j and aP_{task} be the set of available workplaces of *task* on $Knet_j$. According to Equation (2), the probability that wp_j recommends *wp* as the execution place of *task* can be denoted by

$$p^A(wp, task) = \begin{cases} 1 & \text{If } c(wp, task) = \min\limits_{wp' \in aP_{task}}\{c(wp', task)\} \\ 0 & \text{Else} \end{cases} \tag{3}$$

Correspondingly, the instant cost for executing *task* on *wp* is

$$c(wp, task) = wp.sC \tag{4}$$

3. Mobile Agent Path Planning Based on Window Strategy.

3.1. Minimum window cost strategy. According to Definition 2.3, wp_j has two possible strategies to recommend workplace under state s_j at time j. The first strategy is called one-step strategy, which means that while wp_j searches workplace for $task_{j+1}$, it only focuses on the cost of $task_{j+1}$ and does not consider any other tasks remained in s_j. The second strategy is called multi-step strategy, it is one that while wp_j searches workplace for $task_{j+1}$, it not only cares of the cost of $task_{j+1}$ but also the accumulated costs of one or more tasks following $task_{j+1}$ in s_j, which is similar to the "thinking for multi-step before taking one step" behavior in a chess game. Obviously, the one-step strategy is a special case of the multi-step strategy.

To be easier discussion, both one-step strategy and multi-step strategy are called window strategy in the followings.

Let $tW_j = (task_{j+1}, task_{j+2}, \ldots, task_{j+n})$ with $n \geq 1$ be the sequence tasks remained in s_j at time j.

Definition 3.1. *tW_j is called the planning window of the mobile agent under state s_j at time j, and n is called the window width.*

Let $tP = (p_1, p_2, \ldots, p_n)$ be a sequence of available workplaces on $Knet_j$ for tW_j and there are one-to-one mapped relations between tW_j and tP.

Definition 3.2. *The site sequence (wp_j, tP) is called the window path for tW_j, denoted by $wpath = (wp_j, tP)$, and the accumulated costs on wpath is called the window cost, denoted by $C(wpath)$, where wp_j is the current workplace of mobile agent at time j.*

The window cost can be calculated by Formula (5), of which $p_k \in tP$ and $0 \leq \gamma_k \leq 1$ is known as the discount factor,

$$C(wpath) = wp_{j+1}.sC + \sum_{k=1}^{n} \gamma_k * p_k.sC \tag{5}$$

In general, the farther the task was from $task_{j+1}$, the less its contribution to optimal workplace choice is, so γ_k should be in a decreasing order. For simplicity, let all γ_k be the same constant, and Formula (5) can be rewritten as

$$C(wpath) = wp_{j+1}.sC + \gamma \sum_{k=1}^{n} p_k.sC \tag{6}$$

Definition 3.3. *Given tW_j at time j, the window strategy π^w refers to the strategy that wp_j recommends the site p_1 of wpath with minimum window cost to mobile agent as the execution place of $task_{j+1}$. Shown in probability,*

$$p(p_1, task_{j+1}) = \begin{cases} 1 & \text{If } p_1 \in wpath \wedge C(wpath) = \min\limits_{wpath' \text{ on } Knet_j} \{C(wpath')\} \\ 0 & \text{Else} \end{cases} \tag{7}$$

Since there are no failure sites on the mobile agent path, every time the mobile agent moves forward by one workplace, there would be one less task remained in *SBP*. As a result, for time sequence $j = 0, 1, 2, \ldots, N-1$, the maximum width suited for planning window is in a decreasing order.

Property 3.1. *The maximum width suited for tW_j at time j is $N - j$, denoted by $MaxWd_j = N - j$, where N is the total number of tasks in SBP.*

It could be found from Figure 1 that at any time j, the number of tasks in tW_j can not exceed the number of tasks in s_j, that is $MaxWd_0 = N$, $MaxWd_1 = N - 1, \ldots$, $MaxWd_{N-1} = 1$.

3.2. **Construction algorithm of mobile agent path based on window strategy.** At time j, supposing that there are no isolated nodes or isolated sub-networks on $Knet_j$ of wp_j, and all tasks in tW_j can be completed on $Knet_j$. The workplace recommendation process at time j includes four basic steps: (1) the mobile agent sets tW_j and submits it to wp_j for help; (2) wp_j generates its acquaintance network $Knet_j$ based on requirements of tW_j; (3) wp_j finds the $wpath^* = (wp_j, tP)$ with minimum window cost for tW_j on $Knet_j$; (4) wp_j recommends the first site of tP to the mobile agent.

Algorithm 1 Workplace Recommendation Algorithm with Minimum Window Cost
Input to wp_j: $tW_j = (task_{j+1}, task_{j+2}, \ldots, task_{j+n})$ with $1 \leq n \leq N - j$ at time j;
Output to mobile agent: wp_{j+1} for executing $task_{j+1}$ with $c(wp_{j+1}, task_{j+1})$;
Begin

According to requirements of tW_j and Definition 2.4, wp_j generates its $Knet_j$ satisfying the condition that there is at least one available workplace for each task in tW_j;

Let $tWP = \phi$; //tWP is the set of available workplaces for tW_j on $Knet_j$

For $k = 1$ to n Do

wp_j finds the set of available workplaces aP_{j+k} of $task_{j+k} \in tW_j$ on $Knet_j$ and makes $tWP = tWP \cup aP_{j+k}$;

wp_j builds an overlay network above $Knet_j$, denoted by $tWnet = (wp_j, tWP)$;

Let $wPATH = \phi$; //$wPATH$ is the set of window path for tW_j

Repeat

{

wp_j generates a window path for tW_j on $tWnet$, denoted by $wpath$, calculates $C(wpath)$ with Formula (6), and makes $wPATH = wPATH \cup \{wpath\}$;

} Until there is no new window path that could be found on $tWnet$;

$MinC = +\infty$; //wp_j finds the path with minimum window cost

For all $wpath$ in $wPATH$ Do

If $C(wpath) < MinC$ then $\{wpath^* \leftarrow wpath;\ MinC \leftarrow C(wpath^*);\}$

wp_j recommends the first site of tP in $wpath^* = (wp_j, tP)$ to mobile agent as wp_{j+1} for executing $task_{j+1}$ with $c(wp_{j+1}, task_{j+1}) = wp_{j+1}.sC$;

End;

Property 3.2. *For all task $\in tW_j$, if $Knet_j$ was constructed as a referral network and the task can be completed on $Knet_j$, then wp_j is able to find its available workplace with path length no longer than 6 on the average.*

Following the 'Six Degree of Separation' theory from the small world model [19,20], if the available workplace of the *task* exists on $Knet_j$, then wp_j is able to find it with path length no longer than 6 on the average.

Algorithm 2 Path Building Algorithm with Minimum Window Cost
Input to mobile agent: $SBP = (task_1, task_2, \ldots, task_N)$ with $N \geq 1$;
Output to mobile agent: $path$ with cost C_{path};
Begin

At wp_0 where the mobile agent dispatched, the mobile agent sets $path = (wp_0)$ with $C_{path} = 0$ and marks time $j = 0$;

Repeat

{

At time j, the mobile agent sets $tW_j = (task_{j+1}, task_{j+2}, \ldots, task_{j+n})$ with $1 \leq n \leq N - j$ at first, and then submits it to wp_j for help;

Using Algorithm 1, wp_j recommends wp_{j+1} for $task_{j+1}$ with $c(wp_{j+1}, task_{j+1})$ to the mobile agent;

The mobile agent appends wp_{j+1} to $path$, accumulates C_{path}, and migrates to wp_{j+1} to execute $task_{j+1}$;

After $task_{j+1}$ has been completed, the mobile agent marks $j = j + 1$;
} Until $j = N - 1$;
End;

Property 3.3. *With window strategy, the mobile agent is able to dynamically build a minimal cost path for executing sequential tasks.*

According to Property 3.2, as long as tW_j can be completed on $Knet_j$, wp_j can always find wp_{j+1} for $task_{j+1}$ with minimum cost on $Knet_j$ at each time $j = 0, 1, 2, \ldots, N - 1$. As a result, the *SBP* will not be interrupted, and the mobile agent will not either deviate from the direction of the minimum window cost. Therefore, the mobile agent is able to dynamically build a minimal-cost path for *SBP*.

4. Simulation and Example.

4.1. **Simulation experiments.** The purpose of experiments is to examine the impact of window width on the cost of mobile agent path. In order to keep Property 2.4, the simulation environment is constructed by the following steps.

1. Generating the site network, denoted by $WPnet = (WP, E)$, of which *WP* is the set of M points randomly generated on a two-dimension space and each point mapped a site; $\forall u, v \in WP \wedge u \neq v$, $e = (u, v) \in E$ is an edge randomly connected with probability of $\beta \times \exp\left[rand(0,1)\frac{-M}{\mu}\right]$, which represents the acquaintance relation between u and v, where $\mu \geq 1$ is the expected node degree, $0 < \beta < 1$ is a parameter to control the convergence of μ [21].

2. Set $SBP = (task_1, task_2, \ldots, task_N)$ with $N \geq 1$.

(1) Generating the service capability matrix and the service cost matrix on *WP-net*, denoted by $A = [a_{pq}]$ and $C = [c_{pq}]$ respectively, where $p = 1, 2, \ldots, M$, $q = 1, 2, \ldots, N$, of which $a_{pq} = rand(0, 1)$ indicates the service ability of wp_p for $task_q$, and $c_{pq} = rand(C_L, C_U)$ is the service cost of wp_p for $task_q$, where C_L and C_U are the lower and upper bound of the cost respectively.

(2) Generating the capability requirement vector of *SBP*, denoted by $\Theta = (\theta_1, \theta_2, \ldots, \theta_N)$, of which $\theta_q = \frac{1}{M}\sum_{p=1}^{M} a_{pq}$ is the capability requirement of $task_q$.

3. For all $p = 1, 2, \ldots, M$, $q = 1, 2, \ldots, N$, if $a_{pq} \in A \geq \theta_q \in \Theta$ then wp_p is considered as an available workplace of $task_q$, and $task_q$ can be completed automatically.

The process generating a mobile agent path is called one trial and the trial includes two steps:

Step 1. Randomly choose a node from *WPnet* as the dispatching site of the mobile agent, denoted by wp_0;

Step 2. Generate the minimal cost path with Algorithm 1 and Algorithm 2 on *WPnet*, denoted by $path = (wp_0, wp_1, \ldots, wp_N)$.

The trial was repeated for λ times while keep *WPnet* unchanged. Let C_i be the path cost after the ith trial, the average path cost after λ trials is

$$\bar{C} = \frac{1}{\lambda}\sum_{i=1}^{\lambda} C_i \tag{8}$$

Since wp_0 was randomly chosen from *WPnet* in each trial, the trials are independent.

Set $\mu = 10$, $\beta = 0.5$, $C_L = 5$, $C_U = 10$, $N = 20$ and $\lambda = 30$. The experiment results were shown in Figure 3 with different number of nodes M and the different window width n, where $n = 1$ corresponds to the one-step strategy and $n > 1$ corresponds to the multiple-step strategy.

Clearly, as the window width n increases, the path cost $\bar{C}$ decreases. While $n > 3$, the decreasing trend of $\bar{C}$ becomes slow, this is because the more tasks had been taken into

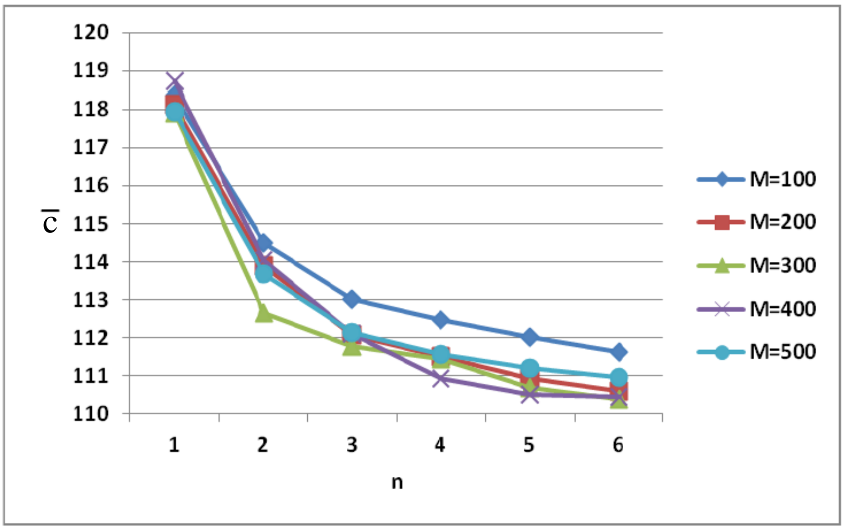


FIGURE 3. The curve of $\bar{C}$ with different M and n

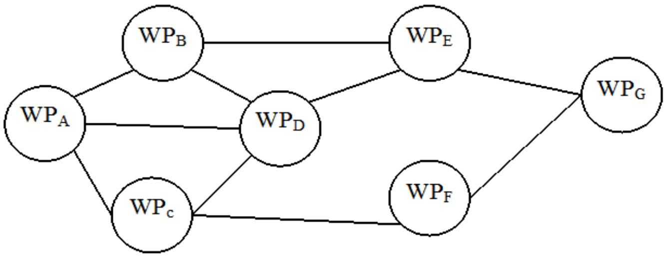


FIGURE 4. An example of acquaintance network of manufacturing alliance

planning window, the higher the probability of nodes repeated by λ trials was with fixed M, so the curve of $\bar{C}$ becomes stable.

Since the service capability and the service cost of sites were both generated randomly in the experiment environment, there is no obvious dominant relation among $\bar{C}$ with different M values.

4.2. **An example.** Figure 4 shows the acquaintance network of manufacturing alliance for a product. The manufacturing alliance is composed of manufacturer $A \sim G$ whose workplace are represented by $WP_A \sim WP_G$. Each workplace provides a set of services, such as part manufacturing and product assembling. It is assumed that the alliance manufactures products based on orders. Clients put forward the manufacturing task of new products by giving orders and product manufacturing requirements to the alliance. A mobile agent can be regarded as the execution proxy of the manufacturing task for this new product. It can migrate to one of the workplaces according to the task, and use the services provided by the workplace to complete a part of manufacturing work of product, by means of the cooperation with a group of workplaces, so as to complete the whole manufacturing task of the new product.

A product manufacturing requirement has an "and/or tree" structure as shown in Figure 5, which does not only describe what parts are included in a product, but also describes the assembly relationship between each part. The workplace where client order request is received will make the business process of the production task based on client order and manufacturing requirement for this product and the goal of the mobile agent is

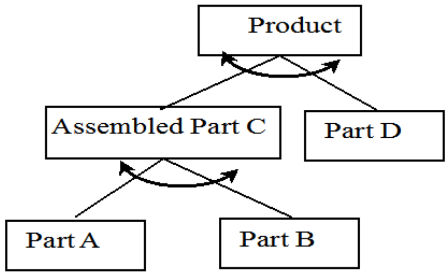


Figure 5. An example of product manufacturing requirement

Table 1. Services providing workplaces for tasks from *SBP* and service costs

Task	Workplace						
	WP_A	WP_B	WP_C	WP_D	WP_E	WP_F	WP_G
t_1	–	6	–	8	–	–	13
t_2	–	–	7	–	12	–	–
t_3	–	–	–	10	–	14	–
t_4	–	–	8	–	–	–	9
t_5	–	6	–	–	9	15	–

to execute this business process. The goal of the mobile agent can be obtained from the product manufacturing requirement. The goal of the mobile agent is obtained from Figure 5. $SBP = (t_1$: manufacturing part A, t_2: manufacturing part B, t_3: assembling part A and part B to assembled part C, t_4: manufacturing part D; t_5: assembling assembled part C and part D to make the product).

Suppose WP_A is the workplace where client orders and product manufacturing requirements are received, WP_A becomes the workplace where the mobile agent launched. No task belonging to *SBP* is executed on WP_A. Table 1 shows the services providing workplaces for tasks from *SBP*, and their service costs.

The data in the table represents the service costs for the task to be executed, while – means the workplace is not able to provide task related services.

Set the window width $n = 1, 2, 3$, the mobile agent path can be built with Algorithm 2. When $n = 1$, the mobile agent path would be $WP_A \to WP_B \to WP_E \to WP_D \to WP_C \to WP_F$, and the path cost would be 51; when $n = 2$, the mobile agent path would be $WP_A \to WP_D \to WP_C \to WP_D \to WP_C \to WP_F$ and the path cost would be 48; when $n = 3$, the mobile agent path would be $WP_A \to WP_D \to WP_C \to WP_F \to WP_G \to WP_E$ and the path cost would be 47. As the window width n increases, the working path fee of the mobile agent reduces as n increases.

5. **Related Work.**

The approaches used to construct mobile agent path include static planning methods and dynamic planning methods. The former refers to methods in which the path was generated before mobile agent launching and the mobile agent possesses complete path knowledge. The later refers to methods in which the mobile agent had no path knowledge before it starts migrating, and it has to dynamically choose workplace through environment exploration.

The most commonly used static planning methods are itinerary method and path learning method. The itinerary method [3,4] concluded the migration behavior of mobile agents as structured patterns, such as selection, and parallel. Since the itinerary patterns lack flexibility, it cannot adapt well to the environment changes. Making the path with redundant sites [9-11] can provide some flexibility for mobile agent. However, as the redundant sites must be settled down before mobile agent migrating, the adaptability to environment changes is limited. The path learning method regards the mobile agent path planning as a machine learning problem on a known network, for instance, TAP algorithm [6,7], genetic algorithm [22] and intelligent swarm algorithm [23-26]. Compared to the itinerary method, the path learning method has the advantage of being able to fully utilize the statistic information of network. However, due to the fact that the network information for learning is the historical information before the mobile agent launched, the path is still not able to well adapt to the environmental changes.

Comparing with the static planning methods, the dynamic planning methods have better adaptability to the open environment. However, due to taking a long time to explore the environment, it has lower efficiency. The most commonly used dynamic planning methods include workplace discovery method and workplace recommendation method. The examples of the workplace discovery method are PHA* algorithm [12], LCF and GCF algorithm [13,14], etc. As it requires sufficient knowledge and capability for workplace discovery, the workplace discovery method leads to a larger size of the mobile agent, and usually results in a high probability of migration delay and migration failure. An example of the workplace recommendation method is the navigation method [27]. Since the navigation method separates the workplace discovery function to the navigation sites, the mobile agent could be light weighted. However, as it requires knowing the whole environment in order to build and maintain the navigation tree, the navigation method does not well support the open environment.

The method given in this paper has the same purpose with the navigation method from the perspective of mobile agent light weighted and workplace recommendation. Comparing with the navigation method, the main advantages of the method given in this paper is that the path planning method given in this paper can dynamically adapt to open environment. Since the workplace recommendation method established in this paper was built on the referral network model but that the whole environment, there is no need to know the whole environment for the workplace recommender. As a result, the path planning method given in this paper fits better with the open environment, especially for dynamic social cooperative work.

In comparison with the 'thinking for one step before taking one step' strategy utilized by almost dynamic planning methods, the 'thinking for multi-step before taking one step' window strategy given by this paper is a partial-global optimization policy in essence. Since the window strategy considered multiple task contributions to explore the optimal workplace, using the window strategy can construct a better mobile agent path than the one-step strategy.

6. **Conclusions.** The major contributions of this paper include: (1) established an MDP-based mobile agent path planning model, which fits well to characterize the mobile agent migration behavior for performing a sequential business process; (2) gave a workplace recommendation method based on the referral network model with window strategy, of which the window path exploration is in favor with partial-global optimization policy, and the referral network model suits for the application in open environments. The simulation results are shown that, the bigger the window width was, the less cost was paid to workplace on the path.

The window strategy provides an available approach to mobile agent path dynamic planning, especially suits for social cooperative works, such as supply chain management, electronic commerce, and tourist service reservation. Following this paper, what needs to be done in the future includes the path planning method with window strategy and time constraints, the path planning method for multiple cooperative mobile agents based on window strategy, the mobile agent path planning method with credible workplace selection policy, etc.

Acknowledgments. We would like to thank the reviewers for carefully reading our manuscript and giving detailed comments and suggestions that have been helpful to improve the manuscript. The work presented this paper is supported by the National Science Foundation of China (Grant No. 60573169, No. 61173068) and the Natural Science Foundation of Shandong Province of China (Grant No. ZR2009GM021).

REFERENCES

[1] R. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, New Jersey, 1957.
[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.
[3] Y. Aridor and D. B. Lange, Agent design patterns: Elements of agent application design, *Proc. of the 2nd International Conference on Autonomous Agents*, pp.108-115, 1998.
[4] D. Chacón, J. McCormick, S. McGrath and C. Stoneking, Rapid application development using agent itinerary patterns, *Technical Report #01-01*, Lockheed Martin Advanced Technology Laboratories, 2000.
[5] S. W. Loke, H. Schmidt and A. Zaslavsky, Programming the mobility behavior of agents by composing itineraries, *Lecture Notes in Computer Science*, vol.1742, pp.214-226, 1999.
[6] K. Moizumi, *Mobile Agent Planning Problem*, Ph.D. Thesis, Dartmouth College, Hanover, NH, 1998.
[7] K. Moizumi and G. Cybenko, The traveling agent problem, *Mathematics of Control, Signals and Systems*, vol.14, no.3, pp.213-232, 2001.
[8] J. W. Baek, G. T. Kim and H. Y. Yeom, Timed mobile agent planning for distributed information retrieval, *Proc. of the International Conference on Autonomous Agents*, pp.120-121, 2001.
[9] K. Koukoumpetsos and N. Antonopoulos, Agent navigator: A model for dynamic, flexible agent mobility, *Proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications*, 2000.
[10] J. W. Baek, J. H. Yeo and H. Y. Yeom, Agent chaining: An approach to dynamic mobile agent planning, *The 22nd IEEE International Conference on Distributed Computing Systems*, pp.579-586, 2002.
[11] D. Y. Liu, B. Yang, K. Yang and S. S. Wang, Migration strategies of mobile agent based on itinerary graph, *Journal of Computer Research and Development*, vol.40, no.6, pp.838-845, 2003.
[12] A. Felner, R. Stern, S. Kraus and A. Ben-Yair, PHA*: Finding the shortest path with A* in an unknown physical environments, *Journal of Artificial Intelligence Research*, vol.21, pp.631-679, 2004.
[13] H. Qi and F. Wang, Optimal itinerary analysis for mobile agents in ad hoc wireless sensor networks, *Proc. of the IEEE 2001 International Conference on Communications*, pp.147-153, 2001.
[14] M. Chen, V. Leung, S. Mao, T. Kwon and M. Li, Energy-efficient itinerary planning for mobile agents in wireless sensor networks, *Proc. of the IEEE 2009 International Conference on Communications*, pp.1-5, 2009.
[15] P. E. Hart, N. J. Nilsson and B. Raphael, A formal basis for the heuristic determination of minimum cost paths, *IEEE Transactions of Systems Science and Cybernetics*, vol.4, no.2, pp.100-107, 1968.
[16] X. L. Wang and G. Z. Zeng, A dynamic planning method for mobile agent implementation based on service recommendation, *Applied Mathematics & Information Sciences*, vol.8, no.3, pp.1247-1255, 2014.
[17] X. L. Wang, G. Z. Zeng and X. S. Xu, A dynamic building method of mobile agent path based on referral networks, *Chinese Journal of Electronics*, vol.23, no.1, pp.1-5, 2014.
[18] P. H. Reingen and J. B. Kernan, Analysis of referral networks in marketing: Methods and illustration, *Journal of Marketing Research*, vol.23, no.4, pp.370-378, 1986.
[19] B. A. Nardi, S. Whittaker and H. Schwarz, It's not what you know, it's who you know: Work in the information age, *First Monday*, vol.5, no.5, pp.1-26, 2000.
[20] S. Milgram, The small world problem, *Psychology Today*, vol.1, no.1, pp.61-67, 1967.
[21] K. J. Liu, Z. W. Yu and Z. Q. Cheng, Study on model for stochastic network simulation and its key techniques, *Computer Engineering and Applications*, vol.41, no.14, pp.130-132/217, 2005.

[22] S. H. Piao and B. R. Hong, A path planning approach to mobile robot under dynamic environment, *ROBOT*, vol.25, no.1, pp.18-21/43, 2003.
[23] J. Jiao, S. Yao and C. Xia, Application for artificial bee colony algorithm in migration of mobile agent, *Communications in Computer and Information Science*, vol.93, pp.232-238, 2010.
[24] Y. W. Feng, Z. Q. Wei and X. P. Ji, A MA sub-optimal migration strategy based on probability, *Journal of Harbin Institute of Technology*, vol.40, no.11, pp.1815-1819, 2008.
[25] J. Li, Y. Chai, P. Li and H. Yin, Multi-cellular-ant algorithm for large scale capacity vehicle route problem, *Advances in Swarm Intelligence, Lecture Notes in Computer Science*, vol.6728, pp.260-266, 2011.
[26] J. Cheng and G. Z. Zeng, A greedy particle swarm optimization algorithm for workplace planning in migrating workflow, *The 4th International Conference on Natural Computation*, vol.6, pp.407-411, 2008.
[27] J. Cheng, G. Z. Zeng and H. He, Research on migrating instance path planning in migrating workflow system, *Journal of Frontiers of Computer Science and Technology*, vol.2, no.6, pp.658-665, 2008.