Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/080339

A Dynamic Planning Method for Mobile Agent Implementation based on Service Recommendation

Xiaolin Wang[*] *and Guangzhou Zeng*

School of Computer Science and Technology, Shandong University, Jinan 250101, P. R. China

Received: 10 Jun. 2013, Revised: 15 Oct. 2013, Accepted: 16 Oct. 2013
Published online: 1 May. 2014

Abstract: Mobile agent is a task proxy of the social member that is able to migrate continuously among hosts and use the local services to complete tasks. The hosts, known as workplaces, can be regarded as service proxies of other social members. The sequence of workplaces on which mobile agent completes its tasks is called path. The main contribution of this paper is proposed a dynamic planning method for mobile agent path based on the variable decision space. The path planning model is described with MDP, and the variable decision space is generated through social acquaintance recommendation, assuming that every social member can provide workplace and service recommendation for mobile agent in cooperative work manner. The experimentation results show that by utilizing social acquaintance recommendation, mobile agent is able to dynamically construct its path based on variable recommended space.

Keywords: Mobile agent, dynamic path planning, markov decision process, social acquaintance recommendation, variable decision space.

1 Introduction

Mobile agent is a task proxy of the social member that is able to migrate continuously among hosts and use local service of hosts to perform tasks. The hosts, known as workplaces, can be regarded as service proxies of other social members. The sequence of workplaces on which mobile agent completes tasks is called path. Path planning has been one of the key issues in mobile agent application.

Mobile agent path planning methods can be divided into static planning and dynamic planning [1,2]. The former refers to approaches that the path has been created before mobile agent launching. The latter refers to approaches that mobile agent has no information about the path before launching, and the workplace must be dynamically decided at every moving stage based on task to be executed and knowledge of environment.

The most commonly used static planning method is itinerary design [3,4,5]. Since information of workplaces on the path is the prior knowledge about environment, it cannot adapt well to the environment changes, such as network disconnection, service change, host failure, etc.

Reference [6,7,8,9,10] gave a kind of dynamic path planning methods based on service discovery, in which mobile agent detects environment at first, and then evaluates and selects workplace every time it migrates. As compared with itinerary designing, the service discovery mechanism has more flexibility to environment, while it requires mobile agent to carry sufficient knowledge and coded function to service discovery. Therefore it leads to a larger size of mobile agent itself. As well known, the larger the mobile agent is, the longer migrating time it takes, and the migration failure probability will increase.

Considering the lightweight of mobile agent, reference [11] proposed a navigation method for mobile agent path planning. The main idea of navigation mechanism is dividing the whole environment into several navigating domains, where a navigating station is located at each domain and all the navigating stations are organized as a navigating tree. By tree style navigating, mobile agent is able to dynamically plan its path. However, prior knowledge about the whole environment is necessary for building the navigating tree, therefore this approach cannot adapt well to open environment.

Continuing the previous work [12], this paper gives a dynamic planning method of mobile agent path based on MDP (Markov Decision Process) and social acquaintance recommendation, assuming that every social member can provide workplace and acquaintance recommendation for

[*] Corresponding author e-mail: xlwang@sdu.edu.cn

mobile agent in accordance with their cooperative contracts. Acquaintance recommendation mechanism is similar to the concept of navigation, but it is more suitable for mobile agent working in an open environment, since it does not rely on the prior knowledge about the whole environment.

The structure of the following parts of this paper is: Section 2 gives the definition of mobile agent path; in Section 3 the MDP model of path planning is built; Section 4 discusses the social acquaintance recommendation mechanism; Section 5 describes the path planning algorithms; Section 6 illustrates the experiment results; Section 7 is about discussion on related work; a summary and a prospect about future work is shown in the final section.

2 Definition of mobile agent path

To simplify the discussion, this paper prescribes: (1) $Tspec = (task_i)_{i=1}^{N}$ is a sequence of tasks[1]; (2)$\forall task_i, task_{i+1} \in Tspec$, $task_i$ and $task_{i+1}$ are executed at two adjacent workplaces wp_i and wp_{i+1}respectively[2]; (3) wp_0 is a special place where mobile agent was launched.

Definition 2.1 Mobile agent is defined as $MA = (aid, Tspec, aLog, aEng)$, where

- aid is the mobile agent identifier;
- $Tspec$ is a sequence of tasks and $\forall task \in Tspec$ $task = (tid, InD, OutD, ExecF, MaxPay)$, where tid is the task name; InD is the data set to be served by workplace; $OutD$ is the output data set after the task has been completed; $ExecF$ is the set of functions to be served by workplace; $MaxPay$ is the maximum allowable payment to service;
- $aLog$ is the work log recording the migrating, cooperating and security events;
- $aEng$ is the coded engine including task scheduling, service using, migration, security protection, etc.

Definition 2.2 Workplace is defined as $wp = (pid, Rservs, Tservs, pEng)$, where

- pid is the workplace identifier;
- $Rservs$ is the runtime services for mobile agent, such as migration, communication, service recommendation, etc;
- $Tservs$ is the task services $\forall ws \in Tservs$ and $ws = (sid, ServF, ServD, ServC)$, sid is the service name; $ServF$ is the service functions; $ServD$ is the service data; $ServC$ is the service fees defined on $ServF$ and $ServD$ according to local service rules;

[1] If the process includes parallel tasks, we can divide the process into a group of branches and each branch is only consisting of sequential tasks.

[2] If two adjacent tasks can be finished at the same workplace, we should combine them as one in the process definition stage.

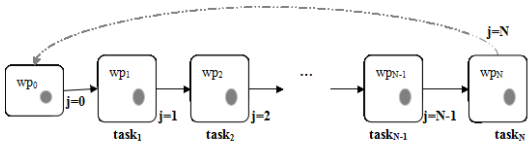

Fig. 1: The path is $wp_0 \rightarrow wp_1 \rightarrow wp_2 \rightarrow ...wp_{N-1} \rightarrow wp_N$

- $pEng$ is the service engine, which provides runtime services and task services for mobile agent as required.

Let WP denote the set of all workplaces in the environment.

Definition 2.3 Given $task \in Tspec$ and $wp \in WP$, wp is an available workplace for $task$ if and only if $\exists ws \in Tservs$ can make $task.ExecF \subseteq ws.ServF \wedge task.InD \subseteq ws.ServD \wedge task.MaxPay \leq ws.ServC$ being true.

Definition 2.4 For $Tspec = (task_i)_{i=1}^{N}$, the mobile agent path is defined as $path = (wp_j)_{j=0}^{N}$, where wp_0 is the launching and returning place of mobile agent. Except for wp_0 , wp_i is the only workplace where $task_i$ can be finished, $i = 1, 2, ..., N$.

Fig.1 gives an illustration of definition 2.4, where $j = 0, 1, 2, N$ indicate the order while mobile agent leaves $wp_0, wp_1, ..., wp_N$ respectively. In convention, the leaving order is also called the migrating-out time and wp_j is named as the current workplace of mobile agent at time j.

Let $g_{agent}^{task_i}$ be the expected gain of $task_i$, $c_{wp_i}^{task_i}$ be the cost executing $task_i$ on wp_i, $c_{wp_i}^{wp_{i-1}}$ be the cost moving from wp_{i-1} to wp_i. The one-step gain of mobile agent to move from to can be represented by

$$r(wp_i|wp_{i-1}) = g_{agent}^{task_i} - c_{wp_i}^{task_i} - c_{wp_i}^{wp_{i-1}} \quad (1)$$

The total gain on the path is

$$G_{path} = \sum_{i=1}^{N} r(wp_i|wp_{i-1}) \quad (2)$$

According to Equation (2), the objective of path planning is to maximize G_{path} through workplace choice at each migrating-out time.

3 Path planning model based on MDP

Let Ω_j be the decision space of mobile agent at migrating-out time j with $wp_j \notin \Omega_j$, where wp_j is the current workplace of mobile agent.

Definition 3.1 The process of mobile agent path planning can be defined as a 4-tuple (S, A, P, R), where

- $S = \{s_0, s_1, ..., s_N\}$ is the set of states, $s_j \in S$ indicates mobile agent prepares to leave wp_j at time j;
- $A = \{a_1, a_2, ..., a_M\}$ is the set of actions, $a_j \in A$ indicates mobile agent evaluates $wp \in \Omega_j$ and decides whether to choose it as wp_{j+1} at time j;

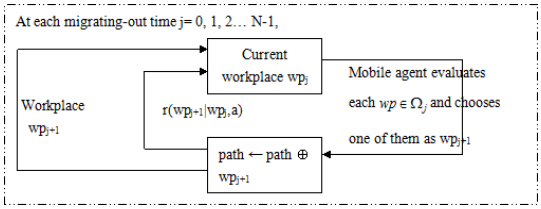

Fig. 2: Path planning model based on MDP. The symbol $\oplus$ represents the linking computation of workplaces.

– $P: S \times A \times S \to [0,1]$ is the state transfer function, $p(s'|s,a) = p(wp|wp_j,a)$ indicates the probability of choosing wp as wp_{j+1} by taking the action a.
– $R: S \times A \times S \to R$ is the reward function, indicates the one-step gain of obtained on wp by taking the action a.

Fig.2 is an illustration of definition 3.1. At each migrating-out time j, mobile agent evaluates all $wp \in \Omega_j$ and chooses one of them as wp_{j+1}.

According to Definition 3.1 and Equation (2), the objective of path planning is to find the optimal policy $\pi^*: S \to A$ to make sure that mobile agent can get the maximum total gain by choosing workplaces from Ω_j at each migrating-out time j, except for $j = N$.

In case of all $\Omega_j = WP$ and WP is known, the path planning can be solved with learning method, such as TAP [13,14,15]. While in a system with variable $\Omega_j \subseteq WP$ and WP is an open environment, it becomes more difficult to use learning method. Therefore we use one-step greedy policy in this paper, that is $\forall j = 0,1,2,\ldots,N-1$, the probability of mobile agent to choose $wp \in \Omega_j$ as wp_{j+1} at wp_j can be expressed as

$$p(wp_{j+1} = wp|wp_j,a) = \begin{cases} 1 & \text{if } r(wp|wp_j,a) = \underset{wp' \in \Omega_j}{MAX}\{r(wp'|wp_j,a)\} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

The method to build the variable Ω_j based on social acquaintance recommendation was given in the next section.

4 Method to build variable decision space

Suppose that every social member in the real world, whatever organization or individual,is rational and can be labeled by element of $\Sigma = 1,2...M$. We use $1,2,\ldots,M$ to identify social members. The social role theory [16] shows that every social member plays a social role and possesses social acquaintance relationships consistent with the role, such as client-servant relationship, etc.

Denoting the acquaintance relationship between members i and j as $SAR(i,j)$ and it is considered as a symmetric relationship.

Property 4.1 The acquaintance relationship has the following properties:

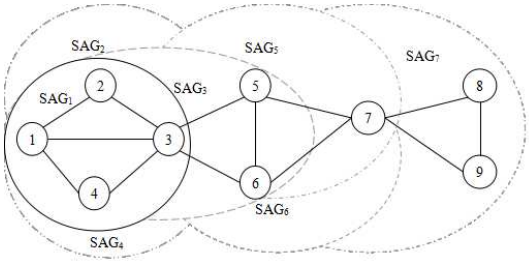

Fig. 3: An example of social acquaintance network and social acquaintance domain.

1. Randomness. Since there is uncertainty in the occurrence of cooperative events, $SAR(i,j)$ is a random variable.
2. Convergence. If $SAR(i,j) \wedge SAR(i,k)$ then the member i has both direct acquaintances j and k.
3. Transitive. If $SAR(i,j) \wedge SAR(j,k)$ then the member i has indirect acquaintance k.

Definition 4.1 Social acquaintance network is defined as $SOA\text{-}Net = (\Sigma, E)$, where Σ is the set of social members, $i \in \Sigma$ represents the social member i; E is the set of edges, $e(i,j) \in E$ shows the direct acquaintance relationship $SAR(i,j)$.

Property 4.2 *SOA-Net* is a random network, and there are no isolated nodes or sub-networks.

Since $SAR(i,j)$ is random, $e(i,j)$ is random and *SOA-Net* is a random network. The assumption of rational social members implies that there is no isolated node in *SOA-Net*, and the convergent and transitive properties of acquaintance relationship ensure that there is no insolated sub-network in *SOA-Net*.

Definition 4.2 For $i \in SOA\text{-}Net$, $SAG_i = \{x\}|_{e(i,x) \in SOA\text{-}Net}$ is an acquaintance group of i, and naming i as the master of SAG_i, x as the group member of SAG_i.

Acquaintance group is the result of the convergence of acquaintance relationship. The degree of node i in *SOA-Net*, denoted as δ_i, equals to the number of group members of SAG_i. Fig.3 is an example of acquaintance groups.

Definition 4.3 Let $i \in SOA\text{-}Net$ be the current service member of mobile agent and $task \in Tspec$ to be executed, direct acquaintance recommendation refers to the process that i recommends service members in SAG_i for *task*, while indirect acquaintance recommendation refers to the process that $x \in SAG_i$ is entrusted to recommend service members in $SAG_x - \{i\}$ for *task* with $SAG_x \not\subset \{i\} \cup SAG_i$, and naming x as the trustee.

Take Fig.3 as an example, here $SAG_1 = \{2,3,4\}$. Since $SAG_2 = SAG_4 = \{1,3\} \subset \{1\} \cup SAG_1$, there is no new member in *SOA-Net* that can be recommended by 2 and 4, they do not have to become trustee of 1. As $SAG_3 = \{1,2,4,5,6\} \not\subset \{1\} \cup SAG_1$, 3 can be entrusted to recommend service members in $SAG_3 - \{1\}$. Because

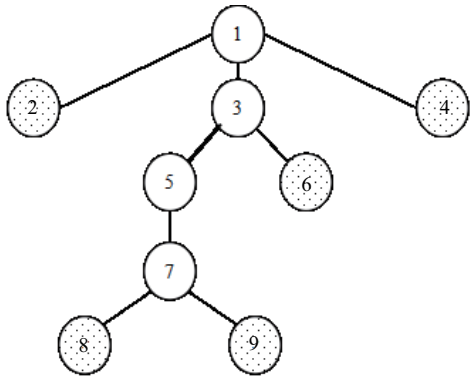

Fig. 4: Acquaintance recommendation tree based on Fig.(3).

two adjacent tasks in $Tspec$ will not be executed in the same workplace, as a result, 3 do not need to check the service of 1 for the task.

Property 4.3 Indirect acquaintance recommendation can be transitively executed in *SOA-Net*.

Since every trustee may become a new master due to the transitive property of acquaintance relationship, indirect acquaintance recommendation can be transitively executed in *SOA-Net*.

Taking example of Fig.3, since $SAG_5 = \{3,6,7\} \not\subset \{3\} \cup SAG_3$ and $SAG_6 = \{3,5,7\} \not\subset \{3\} \cup SAG_3$, 3 is able to pass the task to 5 and 6. However, entrusting duplication will happen if the task is passed from 5 to 6 or from 6 to 5. Similarly, it will result in the same thing if both 5 and 6 pass task to 7. To avoid entrusting duplication, we adopt tree policy in this paper. An acquaintance recommendation tree corresponding to Fig.3 is shown in Fig.4, in which the root represents the current service member 1, the leaf for the members who do not need to be entrusted, and the middle nodes for the members who are both trustee and masters.

Let $i \in SOA\text{-}Net$ be the current service member of mobile agent, the acquaintance tree of i is denoted as$ATree_i$, where i is the root node, and $\forall x \in SAG_i$, x is the leaf node.

Definition 4.4 Given $i \in SOA\text{-}Net$, the acquaintance recommendation tree of i, denoted as *RecTree*, is a tree recursively defined in the following way,

1. $ATree_i$ is a Rec$Tree$;
2. Suppose x is a leaf node on Rec$Tree$, $ATree_x$ is a sub-tree on *RecTree* if and only if
   (a) $\forall y \in ATree_x$, if $y \in$ Rec$Tree$ then cut y from $ATree_x$, and
   (b) The set of leaf nodes $ATree_x$ is not empty.

Definition 4.5 Let $i \in SOA\text{-}Net$ is the current service member of mobile agent and $task \in Tspec$ is the task to be executed, the service recommendation refers to the process that i recommends service members in SAG_i for $task$ at first, and then transitively executes indirect acquaintance recommendation along Rec$Tree$.

For $task \in Tspec$, let Ω_{task} be the set of members created through service recommendation.

Property 4.4 $\forall task \in Tspec$, Ω_{task} varies with task.

According to Definition 4.2, $\forall i \in SOA\text{-}Net$, δ_i varies and group members in SAG_i are distributed unevenly in *SOA-Net*. As a result, Ω_{task} varies with task. The more workplaces there is in Ω_{task}, the higher probability mobile agent are able to find the optimal service.

Theorem 4.1 Given $Tspec$, mobile agent is able to dynamically construct path by way of service recommendation.

Proof: $\forall task \in Tspec$, if $task$ can be completed in social world Σ, there must be $\Omega_{task}(\neq \Phi)$ in *SOA-Net*. Since there is no isolated node or sub-network in *SOA-Net*, if Ω_{task} exists, it must be able to be found via service recommendation. Thus, mobile agent can choose service members from Ω_{task} definitely. That is, mobile agent is able to dynamically construct its path.

$\forall i \in \Sigma$, Supposing that i can provide one and only one workplace wp_i for mobile agent, *SOA-Net* can be mapped into a homogeneous workplace network.

Definition 4.6 Workplace network is defined as $WP\text{-}Net = (WP, E)$, WP is the set of workplaces, $i \in SOA\text{-}Net \rightarrow wp_i \in WP\text{-}Net$; E is the set of edges, $e(i,j) \in SOA\text{-}Net \rightarrow e(wp_i, wp_j) \in WP\text{-}Net$.

According to Definition 4.6, *WP-Net* inherits all properties of *SOA-Net*. In the following discussion, the mapped concepts will be used without redefining.

5 Path planning algorithm

Let wp be the current workplace of mobile agent and $task$ is the task to be executed. Path planning is done in two steps: (1) Mobile agent inquires wp to recommend decision space Ω_{task}; (2) Mobile agent chooses the optimal one from Ω_{task} as new workplace.

Since *WP-Net* is a huge and open network, considering the execution efficiency, generally mobile agent will not request wp to recommend all the service-satisfied workplaces in *WP-Net*. In order to control the recommendation process, let λ be the minimum number of workplaces for all Ω_{task}.

Let $ATree_{wp}$ be the acquaintance tree of wp, δ be the number of leaf nodes on $ATree_{wp}$, Rec$Tree$ be the recommendation tree of wp, and initializing $RecTree = Atree_{wp}$,$\Omega_{task} = \Phi$.

The basic idea of algorithm 1 is: Firstly, wp recommends all the workplaces that can satisfy $task$ in SAG_{wp}. If $|\Omega_{task}| < \lambda$, wp will pass the $task$ to its group members along *RecTree*, and each trustee will recommend all the workplaces that can satisfy $task$ in their own acquaintance group, ..., until $\Omega_{task} \geq \lambda$.

Algorithm 1: Decision space creating algorithm $REC\_WP(wp, task, \Omega_{task}, RecTree)$

```
begin
    XQ ← the sequence of leaf nodes (x1, x2, ..., x_δ) on ATree_wp;
    Ω_task ← All workplaces satisfying in XQ ∪ Ω_task ;
    if |Ω_task| ≥ λ then
        return
    end
    for k ← 1 to δ do
        x ← XQ[k];
        wp obtains ATree_x;
        for each y ∈ ATree_x ∧ y ≠ x do
            if y ∈ RecTree then
                cut y from ATree_x;
            end
        end
        if the set of leaf nodes ATree_x is not empty then
            wp extends ATree_x on RecTree ;
            wp sends (task, Ω_task, RecTree) to x ;
            x executes REC_WP(x, task, Ω_task, RecTree)
            ;
        end
    end
end
```

If the communication cost in passing task is neglected, the time cost of algorithm 1 would be

$$Time_{\Omega_{task}} = \sum_{j=1}^{n} \delta_j \tag{4}$$

of which, n is the number of workplaces that took part in recommendation process for creating Ω_{task}, δ_j is the number of times that workplace j compared $task$ and services, numerically it equals to the number of group members in SAG_j.

It needs to be pointed out that, service recommendation may cause duplicate evaluation of the same workplace if it is a shared group member of two master workplaces in *WP-Net*. Considering that each social member is an autonomous entity, and its acquaintance knowledge is independent, algorithm 1 didnt exclude duplicate evaluation.

Let g_{agent}^{task} be the expected gain of the task, $MaxG$ be the maximum gain actually obtained from Ω_{task}, and $NextWP \in \Omega_{task}$ be the workplace corresponding to $MaxG$.

The basic idea of algorithm 2 is that mobile agent obtains Ω_{task} firstly, and then chooses the workplace with maximum gain as the new one. If we neglect the time spent in communication to obtain $c_{wp_j}^{task}$ and $c_{wp_j}^{wp}$, the factors affecting the time complexity of algorithm 2 are obtaining Ω_{task} and choosing the optimal workplace from Ω_{task} . The time cost of the former can be calculated with Equation (4), and the number of comparison of the latter is $|\Omega_{task}|$.

Algorithm 2: Workplace selecting algorithm $SELEC\_WP()$

```
begin
    Ω_task ← Φ ;
    RecTree ← ATree_wp ;
    REC_WP(wp, task, Ω_task, RecTree);
    MaxG ← −1;
    NextWP ← '';
    for j ← 1 to |Ω_task| do
        Mobile Agent obtains c_{wp_j}^{task} and c_{wp_j}^{wp},
        wp_j ∈ Ω_task;
        r_j = g_agent^task − c_{wp_j}^{task} − c_{wp_j}^{wp};
        if r_j > MaxG then
            MaxG ← r_j;
            NextWP ← 'wp_j';
        end
    end
    return (MaxG, NextWP)
end
```

6 Experiment and analysis

To simplify the calculation, $\forall task \in Tspec$, assuming that the satisfaction degree of $task$ can be denoted by the expected gain $r_{agent}^{task} = \theta$, where θ is a non-zero constant.

6.1 Arrangement of experiment environment

The experiment environment was set with the following method.

1. Generate *WP-Net*
   (a)Randomly generate a set of M points in the two-dimensional space as $WP = \{wp\}_1^M$.
   (b)$\forall wp_i, wp_j \in WP \wedge i \neq j$,connect wp_i and wp_j in the probability of $\beta \times \exp[rand\,(0,1)\frac{-M}{\mu}]$, where $\mu \geq 1$ is the expected value of degrees of nodes, and $0 < \beta < 1$ is the node connection control parameter[17].
2. Generate Ω_{task}
   (a)$\forall task \in Tspec$ randomly choose λ workplaces but wp in WP-Net as Ω_{task}, wp is the current workplace of mobile agent.
   (b)$\forall wp_k \in \Omega_{task}$ generate a random number $rand[\theta,\ 1.5\theta]$ for wp_k as the gain $r_{wp_k}^{task}$ which mobile agent can obtain at wp_k actually, where θ is the same as in r_{agent}^{task}.

6.2 Total gain and efficiency of path planning

According to Algorithm 2, mobile agent always chooses the workplace with maximum gain from Ω_{task} as the

workplace where *task* is executed, the total gain on the path can be calculated with Equation (5),

$$G = \sum_{i=1}^{N} (r_{wp}^{task_i} = \underset{wp' \in \Omega_{task_i}}{MAX} \{r_{wp'}^{task_i}\}) \quad (5)$$

Omitted the cost of entrusting recommendation, the Equation (6) is used to represent the efficiency of path planning, of which, $Time_{\Omega_{task_i}}$ be calculated with Equation (4). The larger η is, the less efficient of planning is.

$$\eta = \frac{1}{N} \sum_{i=1}^{N} Time_{\Omega_{task_i}} \quad (6)$$

6.3 Experiment results and analysis

The objective of experiment is to examine the effect of the changes of environment parameter M and λ on the total gain G and the path planning efficiency η.

Suppose $N = 20$, $\theta = 5$, $\mu = 5$ and $\beta = 0.5$.

The process of generating path for N tasks is named as one time of experiment. For the given M and λ, all the results are the average values of 30 experiments and all of which are done based on the same *WP-Net*, but Ω_{task} and $r_{wp_k}^{task}(wp_k \in \Omega_{task})$ are generated randomly.

Fig 5. shows G and η as λ changes while M is constant. The larger λ is, the greater G is, the larger η is.

When λ risen, $|\Omega_{task}|$ increased, and the number of workplaces with greater gain increased as well. Since mobile agent always selects the workplace with the greatest gain from Ω_{task}, the G increased, as shown in the left part of Fig.5 . In the right part of Fig.5, when λ became larger, $|\Omega_{task}|$ increased, η also increased.

Fig.6 shows G and η as M changes while λ is constant. The larger M is, the larger η is, while G basically stays constant.

Since Ω_{task} is randomly generated in *WP-Net*, when λ is constant while M increases, the probability of Ω_{task} to be sparsely distributed in *WP-Net* will increase, and the number of workplaces that took part in recommendation process for creating Ω_{task} and the number of times comparing *task* and service increases as shown in the right part of Fig.6. Since Equation (5) is independent from M, if λ stays constant, the workplace gain in Ω_{task} can only be affected by random number $\theta \leq r_{wp}^{task} \leq 1.5\theta$. As shown in the left part of Fig. 6, the average result of 30 experiments makes the curve stable despite some fluctuation.

Fig. 7 shows G and η as M and λ changes. The larger M and λ is, the larger η is. However the total gain on the path becomes stable and close to the maximum one.

As shown in Fig.5 to Fig.7, the key parameter that affects G and η is λ. If the goal of path planning is to pursue maximum gain, λ should be larger. If the goal of planning is to pursue maximum efficiency, λ should be smaller. However if $|\Omega_{task}|$ is too small, the adaptability will reduce due to the lack of services.

7 Related works

The problem of mobile agent path planning can be solved in static or dynamic manner.

Static planning is the approach that path is created before mobile agent has been launched. The most commonly used static planning methods are itinerary design and path learning with known WP. Itinerary design method concludes the migrating behavior of mobile agent as structured reusable itinerary patterns, such as sequence, branch and parallel. Itinerary pattern can be described with itinerary languages [3] or itinerary algebra [4]. Since itinerary patterns cannot cover the ad hoc migrating behavior of mobile agent, the static itinerary cannot meet the environment changes. Although itinerary with redundant workplaces [18,19,20] can provide some flexibility to the workplace selection, but the redundant workplace information is a prior knowledge, the adaptability is still limited. The path learning approach regards the environment was fixed and known, such as TAP [13,14,15] , genetic [21] and swarm [22,23,24,25] algorithms. As compared with itinerary design, the advantage of path learning is to be able to use the whole network information automatically. However the network information used in the learning algorithm is the historical information before mobile agent was launching, as a result, the path generated cannot adapt well to the environment changes during the mobile agent implementation. TAP [13,14,15] also describes the path planning issue as MDP with a fixed and known environment. Since the basic idea of TAP is derived from TSP [26], each workplace appears only once in the path. The MDP model introduced in this paper allows decision space vary for deferent tasks, therefore it brings more flexibility to path planning.

Dynamic planning refers to the approach that mobile agent has no path information before it starts to migrate. Every time it migrates, the new workplace is selected based on its goal and knowledge of the current environment. Compared to static planning, the advantage of dynamic planning is high adaptability to an open environment, while the disadvantage is that it takes a long time to understand the environment, therefore decreasing efficiency. The way of obtaining knowledge of the current environment can be divided into service discovery and service recommendation. Examples for the former are PHA*[6], LCF and GCF [7,8], etc. Since service discovery requires mobile agent to carry sufficient knowledge and coded functions with it, it is possible that mobile agent be large, and migration delay and fault will happen. The example for the latter is navigating mechanism [11]. Compared to the service discovery, navigating can make mobile agent lightweight, but constructing the unified navigating tree requires prior knowledge of whole environment and it cannot support an open environment either.

The acquaintance recommendation method introduced in this paper is similar to the navigation from

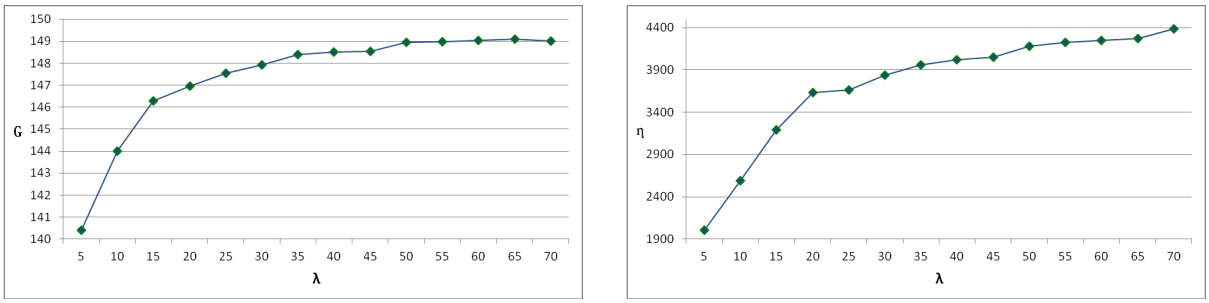

Fig. 5: Changes of G and η as λ increase, $M = 1000$.

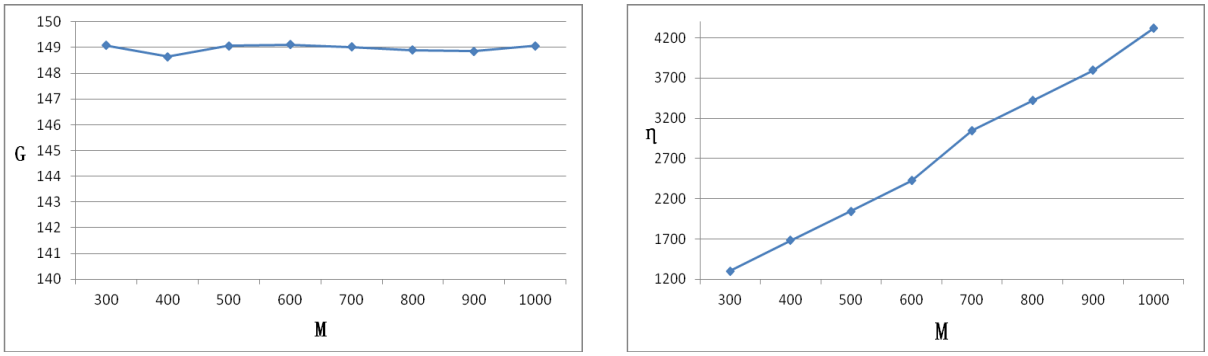

Fig. 6: Changes of G and η as M increase, $\lambda = 60$.

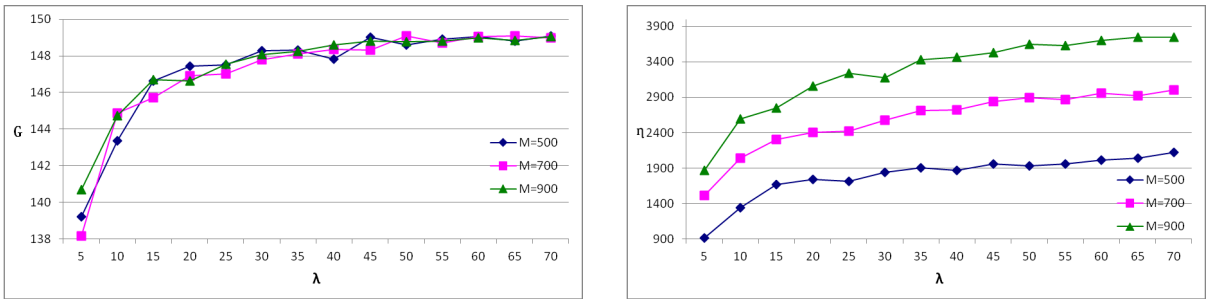

Fig. 7: G and η as M and λ change

the service recommendation perspective. Both of them can make mobile agent lightweight. The acquaintance recommendation mechanism is implemented based on an open social network and does not rely on the whole environment. As long as cooperation between social members is rational, mobile agent can always use acquaintance recommendation to dynamically planning its path.

8 Conclusions

The main contribution of this paper is proposed a dynamic planning method for mobile agent implementation based on the variable decision space. The path planning model is described with MDP, and the variable decision space is generated through social acquaintance recommendation. Since both mobile agent and workplace are proxies of the social members, as long as the goal of mobile agent can be achieved in the social network, mobile agent is constantly be able to dynamically construct its path with the method introduced in this paper. The experiment results and analysis show that the larger the decision space recommended to mobile agent is, the greater the total gain can be obtained, and the less efficient the planning is.

Social acquaintance recommendation provides a feasible way for mobile agent path dynamic planning.

The future work mainly includes: acquaintance recommendation with constrains of time and payment, multi-agent path planning based on social acquaintance recommendation, etc.

Acknowledgement

The work presented this paper is supported by the National Science Foundation of China (Grant No.60573169,61173068) 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, Proceedings of the second international conference on Autonomous agents, Minneapolis, Minnesota, USA, 108-115 (1998).
[4] D. Chacn, J. McCormick, S. McGrath, and C. Stoneking, Rapid Application Development Using Agent Itinerary Patterns, Technical Report #01-01, Lockheed Martin Advanced Technology Laboratories, March, (2000).
[5] S. W. Loke, H. Schmidt, and A. Zaslavsky, Programming the Mobility Behavior of Agents by Composing Itineraries, Lecture Notes in Computer Science, **1742**, 214-226 (1999).
[6] 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, **21**,631-679 (2004).
[7] H. Qi and F. Wang, Optimal Itinerary Analysis for Mobile Agents in Ad Hoc Wireless Sensor Networks, Proceedings of the IEEE 2001 International Conference on Communications, Helsinki, Finland, 147-153 (2001).
[8] M. Chen, V. Leung, S. Mao, T. Kwon, and M. Li, Energy-efficient Itinerary Planning for Mobile Agents in Wireless Sensor Networks, Proceedings of the IEEE 2009 International Conference on Communications, Bresden, Germany, 1-5 (2009).
[9] 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, **4**, 100-107 (1968).
[10] L. Rech, C. Montez, and R. S. Oliveira, Itinerary determination of imprecise mobile agents with firm deadline, An International Journal of Web Intelligence and Agent Systems, **6**, 421-439 (2008).
[11] 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, **2**, 658-665 (2008).
[12] J. L. Gao and G. Z. Zeng, Research of Mobile Agent Workflow Path Dynamic Programming Method Based on Social Acquaintance Network,3rd International Conference on Information Sciences and Interaction Sciences, 398-402 (2010).
[13] K. Moizumi, Mobile Agent Planning Problem, Ph.D. dissertation, Dartmouth College, Hanover, NH, (1998).
[14] K. Moizumi and G. Cybenko, The Traveling Agent Problem, Mathematics of Control, Signals and Systems, **14**, 213-232 (2001).
[15] J. W. Baek, G. T. Kim, and H. Y. Yeom, Timed Mobile Agent Planning for Distributed Information Retrieval, Proceedings of the International Conference on Autonomous Agents, Montreal, Quebec, Canada, 120-121 (2001).
[16] B. J. Biddle, Role Theory: Expectations, Identities, and Behaviors, New York, Academic Press, (1979).
[17] 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, **41**, 130-132/217 (2005).
[18] K. Koukoumpetsos and N. Antonopoulos, Agent NavigatorA Model for Dynamic, Flexible Agent Mobility, Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, Las Vegas, Nevada, USA, (2000).
[19] J. W. Baek, J. H. Yeo, and H. Y. Yeom, Agent Chaining: An Approach to Dynamic Mobile Agent Planning, 22nd IEEE International Conference on Distributed Computing Systems, Vienna, Austria, 579-586 (2002).
[20] 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, **40**, 838-845 (2003).
[21] S. H. Piao and B. R. Hong, A Path Planning Approach to Mobile Robot under Dynamic Environment, ROBOT, **25**, 18-21/43 (2003).
[22] J. Jiao, S. Yao, and C. Xia, Application for Artificial Bee Colony Algorithm in Migration of Mobile Agent, Communications in Computer and Information Science, **93**, 232-238 (2010).
[23] J. Cheng and G. Z. Zeng, A Greedy Particle Swarm Optimization Algorithm for Workplace Planning in Migrating Workflow, Fourth International Conference on Natural Computation, **6**, 407-411 (2008).
[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, **40**, 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, **6728**, 260-266 (2011).
[26] G. Laporte, The Traveling Salesman Problem: An Overview of Exact and Approximate Algorithms, European Journal of Operational Research, **59**, 231-247 (1992).

Xiaolin Wang is currently an associate professor in School of Computer Science and Technology of Shandong University. Her research interests include mobile computing and its application technology, CSCW and workflow management.

Guangzhou Zeng is currently a professor, Ph.D. supervisor. His research interests include CSCW, intelligent computing theory and technology, mobile computing and its application technology.