6th International Conference on Computer Communications and Networks

September 22 -- 24, 1997
Monte Carlo Hotel
Las Vegas, Nevada

# A Novel Mobile Agent Search Algorithm

Wen-Shyen E. Chen

Institute of Computer Science
National Chung-Hsing University
Taichung, Taiwan ROC 40227
echen@cs.nchu.edu.tw

Chun-Wu R. Leng and Yao-Nan Lien

Department of Computer Science
National Chengchi University
Taipei, Taiwan ROC
{leng, lien}@cs.nccu.edu.tw

## Abstract

*Intelligent Agent has been shown to be a good approach to addressing the issues of limited capacity and unreliable wireless links in mobile computing. However, before the approach can be commercially viable, a set of management capabilities that support the controls of intelligent agents in a mobile environment need to be in place. Since controls can only be applied after the target agent is located, an effective agent search algorithm is an indispensable part of the management functions. In this paper, we propose a new algorithm, the Highest Probability First Algorithm, for locating the target agent. The approach makes use of the execution time information to reduce cost and network traffic. The execution time of the agent on a server is assumed to be binomial distributed and therefore is more realistic.*

**Keywords:** Mobile Agent, Management, Agent Location, Search Algorithm

## 1 Introduction

Compared to the conventional computers with a fixed connection to wired networks, mobile computers have narrow, unreliable connectivity, limited processing power and battery capacity, and have to operate in a dynamic, heterogeneous environment [1, 2]. Intelligent Agent [3, 4, 5, 6] is shown to be promising in addressing the issues of limited capacity and unreliable links of mobile computers.

Nevertheless, before an intelligent agent service can be accepted, a high quality and cost effective agent mobility operation, administration and maintenance (OA&M) system must be in place to guarantee a certain level of service quality. For any control to be applied to the target agent, it needs to be located first. Therefore, *agent location* is an indispensable part of the OA&M.

In this paper, we extend the work in [7] and propose a new agent search algorithm, the Highest-Probability-First search (HPFS) algorithm, that makes use of the execution time information. In the HPFS algorithm, the execution time on a server is assumed to be *binomial distributed* [8], which is closer to reality. The derived probability function is shown to be much less complicated and can be adopted by a search agent when being sent to locate the target agent. Although the itinerary of the agent is assumed to be non-branching, the proposed approach can be easily extended to cover the branching cases, as in the timed protocol specification and validation [9, 10].

The rest of the paper is organized as follows. Section 2 describes the Highest-Probability-First search algorithm we propose. Simulation results are presented in Section 3. Concluding remarks and the future research topics will be given in Section 4.

## 2 The Highest Probability First Search Algorithm

### 2.1 Location Estimation

According to the above discussion, the performance of a search algorithm is determined by the time spent on locating the target agent, as well as the network overhead caused by the algorithm. Both evaluation criteria, in fact, are mainly resulted from the number of times that the search agent probes the servers to locate the target agent. Therefore, a strategy of querying to the server with the highest probability among those servers will consequently consume less search time and network overhead than blind search strategies.

The following notations are used in evaluating the HPFS.

1. $(S_1, S_2, \ldots, S_n)$ : an ordered sequence of servers that the target agent will visit. Note that servers $S_i$ and $S_j$ can be the same.

2. $[t'_{S_i}, t''_{S_i}]$ : a an estimated service time range that the target agent stays in server $S_i$. The range can be determined by selecting the worst (widest) time range collected by experiments.

3. $t_{S_i}$ : service time of the target agent completing its job at server $S_i$; i.e., $t''_{S_i} - t'_{S_i}$.

4. $T_{S_i}$ : summation of the service time that the target agent stays in $S_1, S_2$ to $S_i$; i.e., $\sum_{\ell=1}^{i} t_{S_\ell}$.

5. $T'_{S_i}$ : summation of all the minimum service time that the target agent stays in servers $S_1, S_2$ to $S_i$; i.e., $\sum_{\ell=1}^{i} t'_{S_\ell}$.

6. $\bar{\mathcal{F}}^t_{S_i}$ : probability of the target agent still running at server $S_i$ after $t$ seconds since the agent is initially delivered to server $S_i$.

7. $P^T_{S_i}$ : probability that the target agent is currently located at server $S_i$, $T$ seconds after it was initiated at $S_1$.

Instead of blindly searching for the target agent, the HPFS algorithm sends a probe to a server with the highest probability that the agent might currently stay. If the result of probing is negative, the server with the second highest probability will be the next target. This search strategy will continue until the agent is located. Conclusion of the following theorem provides an efficient way to determine the probability values.

**Theorem 1** *Assume that the service time of an agent to complete its job on each server is binomial distributed over the time range* $[t'_{S_i}, t''_{S_i}]$. *After* $T$ *seconds since the target agent is initialized in the first server* $S_1$, *the probability* $P^T_{S_i}$ *of the agent being located in server* $S_i$ *is formulated as:*

$$P^T_{S_i} = \frac{1}{2^{T_{S_{i-1}}}} \sum_{j=0}^{t''_{S_i}} \binom{T_{S_{i-1}}}{T - T_{S'_{i-1}} - j} \times \left(1 - \frac{1}{2^{t_{S_i}}} \sum_{\ell=0}^{t''_{S_i}} \binom{t_{S_i}}{\ell - t'_{S_i}}\right) \quad (1)$$

To verify Theorem 1, without loss of generality, we make the assumption that the probability function, say $\mathcal{F}_{S_i}(x)$, of the service time on server $S_i$ is a *normal-distribution-like* function over the execution time range from $t'_{S_i}$ to $t''_{S_i}$. That is, the agent could spend an arbitrary length of time to finish its work in server $S_i$ within the time range, but the highest probability of the length of time for the agent to complete its job should be around the mid-point between $t'_{S_i}$ and $t''_{S_i}$. The assumption of normal-distribution-like probability function seems to be more practical and reasonable than other distribution function, such as the uniform distribution function. Consequently, the probability function $\mathcal{F}$ can be formulated to be a binomial distribution function, which is a *discrete function* with a shape similar to that of the curvature of a normal distribution function [8].

$$\mathcal{F}_{S_i}(x) = \frac{1}{2^{t_{S_i}}} \sum_{\ell=0}^{t_{S_i}} \binom{t_{S_i}}{\ell} x^{t'_{S_i}+\ell} \quad (2)$$

Note that $x^{t'_{S_i}}$ is the lower bound of the time interval that the agent will stay in server $S_i$. The coefficient of a term $x^{t'_{S_i}+\ell}$ in Eq. 2 represents the probability of agent to spend $t'_{S_i} + \ell$ seconds to accomplish the work in $S_i$. Those which term powers are out of the summation range will be considered to have zero probability. Consequently, Eq. 2 can be simplified as:

$$\mathcal{F}_{S_i}(x) = \frac{1}{2^{t_{S_i}}} x^{t'_{S_i}} \cdot (1+x)^{t_{S_i}} \quad (3)$$

Eq. 3 is mainly used to depict the length of time that agent requires to complete its work in a server. If the time to deliver the agent from the end of a server $S_i$ to the start of the next server $S_{i+1}$ is negligible (or the deliver time can be treated as a part of the responsibility to server $S_i$),

Considering the probability that the target agent is in server $S_i$ after $T$ seconds from the client sending out the agent to the first server, the probability consists of several components. The first component is in conjunction with two probability values: the probability that the previous $i-1$ servers spend all the $T$ seconds services time, and the probability that server $S_i$ will not finish the job with zero second. According to the results of Eq. 3, probability function of the first $i-1$ servers is the production of each individual server probability functions because all the first $i-1$ servers can be considered as one large system, and each individual server among them is just one step of the whole procedure. Therefore, the probability function of the first $i-1$ servers as a whole can be formulated as

$$\prod_{k=1}^{i-1} \mathcal{F}_{S_k}(x) = \frac{1}{2^{T_{S_{i-1}}}} x^{T'_{S_i}} \cdot (1+x)^{T_{S_{i-1}}} \quad (4)$$

Then, the coefficient of the term $x^T$ in Eq. 4 represents the probability that the target agent spends exactly $T$ seconds in $i-1$ servers.

$$\text{coef. of } x^T = \frac{1}{2^{T_{S_{i-1}}}} \binom{T_{S_{i-1}}}{T - T'_S} \quad (5)$$

Next, the probability that the target agent *will not* finish the work at server $S_i$ in $t$ seconds is represented by the notation $\bar{\mathcal{F}}^t_{S_i}$. We can describe the function $\bar{\mathcal{F}}^t_{S_i}$ from a different point of view: the probability value will be one minus each probability value that the job will be done in less than $t$ seconds.

$$\bar{\mathcal{F}}^t_{S_i} = 1 - \frac{1}{2^{t_{S_i}}} \sum_{\ell=0}^{t} \binom{t_{S_i}}{\ell - t'_{S_i}} \quad (6)$$

Concluding from the the discussion above, as well as in Eqs. 5 and 6, Theorem 1 can be verified from the following formulation.

$$P_{S_i}^{T} =$$

$$\frac{1}{2^{Ts_{i-1}}}\binom{Ts_{i-1}}{T-T'_{S_{i-1}}}\times\left[1-\frac{1}{2^{ts_i}}\sum_{\ell=0}^{0}\binom{ts_i}{\ell-t'_{S_i}}\right]+$$

$$\frac{1}{2^{Ts_{i-1}}}\binom{Ts_{i-1}}{T-1-T'_{S_{i-1}}}\times\left[1-\frac{1}{2^{ts_i}}\sum_{\ell=0}^{1}\binom{ts_i}{\ell-t'_{S_i}}\right]+$$

$$+$$

$$\frac{1}{2^{Ts_{i-1}}}\binom{Ts_{i-1}}{T-t''_{S_i}-T'_{S_{i-1}}}\times\left[1-\frac{1}{2^{ts_i}}\sum_{\ell=0}^{t''_{S_i}}\binom{ts_i}{\ell-t'_{S_i}}\right] \quad (7)$$

Note that Eq. 7 is the same as Eq. 1

### 2.2 The Highest Probability First Search Algorithm

With the results from the previous subsection, we propose to include the following Highest Probability First Search Algorithm in the search agent to locate the target agent.

**Highest Probability First Search Algorithm**

```
Main {
    SS = {S_1, S_2, ..., S_n}
    PSL = Sort(SS)
    HPFS (target, SS)
}
Procedure Sort(SS){
    Sort SS in decreasing order according to
    corresponding probability values
}

Procedure HPFS (target, SS){
    if (SS = ∅)
    then return NOT_FOUND
    S_t = First server in PSL
    Move the Search Agent to S_t
    if (target found in S_t)
    then return (S_t)
    else if (S_t has been visited by the target agent)
    then PSL = Sort (SS - {S_1, S_2, ..., S_t})
    else PSL = Sort (SS - {S_t, S_{t+1}, ..., S_n})
    return (HPFS (target, PSL))
}
```

Note that "SS" and "PSL" are global variables.

In this algorithm, if the search agent arrives at a server $S_t$ and finds that the target agent has moved away from that server, then we should exclude all the servers that proceed $S_t$ from the search list. This is based on the assumption that the servers will be visited in sequence. On the other hand, if the target agent has not arrived at $S_t$, then all the servers following $S_t$ in the original execution order will be excluded in the future search for the same reason. The search list will then be sorted according to the servers' corresponding probability values. Since the target agent is still mobile before being located, it is possible that the target agent might "slip through" the search, i.e., the target agent might move to servers excluded from the search list in previous rounds of search. A simple solution is to leave some information at the servers that the search agents have visited and ask the target agent to report its position and status when it arrives at those servers.

## 3 Simulation Results

In this section, we present the simulations results and use "number of probes" needed to locate the target agent as a performance measure to compare the basic binary search and the HPFS.

In the simulations, we assume that the target agent will visit twenty servers, numbered in sequence form 1 to 20. The service time ranges for the servers are as shown in Table 1. The elapse times range from 1 to 200 in the simulations and the service time range for server 20 is chosen so that it can be a "sink", i.e., the target agent will not go beyond server 20. The values of the simulation results are obtained by taking the average of the results of 1000 runs.

Table Service Time Ranges for the Servers

| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| $t'_{S_i}$ | 3 | 4 | 6 | 2 | 3 | 2 | 8 | 7 | 4 | 1 |
| $t''_{S_i}$ | 8 | 20 | 17 | 10 | 14 | 6 | 16 | 22 | 20 | 9 |

| S | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|
| $t'_{S_i}$ | 8 | 6 | 3 | 5 | 9 | 2 | 4 | 2 | 7 | 500 |
| $t''_{S_i}$ | 17 | 20 | 18 | 14 | 13 | 11 | 17 | 21 | 28 | 600 |

Our goal is to predict with certain accuracy where the target agent is when the elapse time and execution time ranges are given. Therefore, it is of interest to know if the Eq. 1 can show the probability of where the agent is accurately and if the the difference between the theoretical and simulations results vary with elapse time. Figure 1 that the simulation results are very close to the theoretical results, with the highest probability all pointing to the same server.

Figure 2 shows the comparison of the basic binary search and the HPFS algorithms. As illustrated in the figure, the basic binary search algorithm needs more

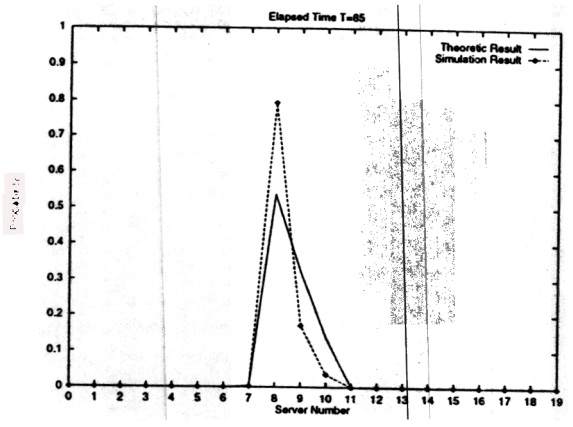


Figure 1: Probability Distribution of the Location of the Agent (Elapse Time = 85).

probes to locate the target agent. In addition, the numbers of probes needed vary in a wide range with the elapse time. On the contrary, the expected values of the probes needed for the HPFS algorithm are lower than those for basic binary search and the variation is much smaller. The simulation results match the theoretical results quite well. Consequently, the validity of Theorem 1 can be verified.

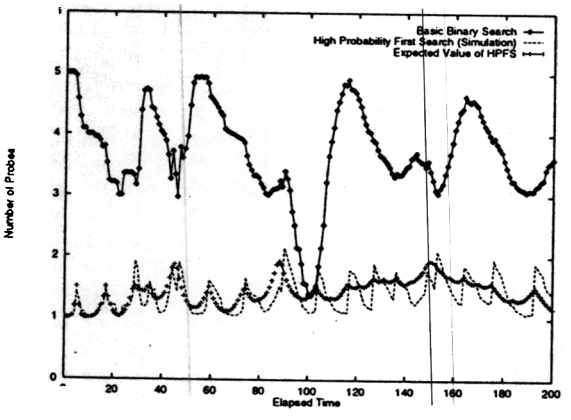


Figure 2: Comparison of the Basic Binary Search and the HPFS.

## 4 Conclusion

Agent location is an indispensable part of the management functions needed to support the mobility of intelligent agents. Straightforward agent search algorithms often lead to excessive network traffic. In this paper, we have proposed a new agent search algorithm that makes use the execution time information available. The execution time is assumed to be binomial distributed, which is closer to reality. With the probability functions we came up with, the Highest Probability First Search algorithm is then formulated. It is expected to generate less probes to locate the target agent. The simulation results match the theoretical results quite well

We had made the assumption that the target agent traverse the servers in a predetermined order. However, the path the target agent takes might depend on the real time condition and could be nondeterministic. In addition, the relationship between agent location and agent control functions (how to apply the control function after the target agent is located?) needs to be clarified. We plan to resolve these problems in the future research.

## Acknowledgment

We gratefully acknowledge the help of C.-Y. Lin, Y.-T. Liu, and C.-J. Chu for writing the simulation programs.

## References

[1] G. H. Forman and J. Zahorjan. The Challenges of Mobile Computing. *IEEE Computer*, pages 38–47, March 1994.

[2] T. Imielinski and B. R. Badrinat. Mobile Wireless Computing: Challenges in Data Management. *Communication of the ACM*, August 1994.

[3] M. Wooldridge and N. R. Jennings, editors. *Intelligent Agents: Theories, Architectures, and Languages.* Springer-Verlag Lecture Notes in AI - vol. 890, 1995.

[4] M. Wooldridge and N. R. Jennings, editors. *Intelligent Agents II: Theories, Architectures, and Languages.* Springer-Verlag Lecture Notes in AI - vol. 1037, 1996.

[5] N. Jennings and M. Wooldridge. Software Agent. *IEEE Personal Communications Magazine*, pages 17–20, Janurary 1996.

[6] W.-S. E. Chen and Y.-N. Lien. Intelligent Messaging for Mobile Computing over the World-Wide Web. In *Proceedings of the Second Workshop on Mobile Computing*, April 1995.

[7] Y.-N. Lien and C.-W. R. Leng. On the Search of Mobile Agents. In *Proceedings of the 7th IEEE Symp. of Personal, Indoor, and Radio Communications*, October 1996.

[8] R. Jain. *The Art of Computer System Performance Analysis.* John Wiley & Sons, Inc., 1991.

[9] F.-J. Lin, P. M. Chu, and M. T. Liu. Protocol Verification Using Reachability Analysis. *ACM Computer Communicaiton Review*, 17(5):126–135, 1987.

[10] G. J. Holzmann. *Design and Validation of Computer Protocols.* Prentice-Hall, 1991.