ANALYSIS OF MOBILE AGENT SYSTEMS PERFORMANCE USING LINEAR PREDICTION MODEL

1FAIZ AL-SHROUF, 2AIMAN TURANI
1Assoc. Prof., Amman Arab University, Faculty of Computer Sciences and Informatics, Amman-Jordan
2Assoc. Prof., Applied Science University, Faculty of Computer Engineering, Amman-Jordan
E-mail: 1faiz_alshrouf@aau.edu.jo, 2aimant@asu.edu.jo

ABSTRACT

Mobile Agents are considered an interesting technology to develop applications for distributed systems. Thus they present features, such as autonomy and capability to roam to hosts, process data, and save remote communications. Many mobile agent platforms have been developed for research purposes while other platforms have been deployed as commercial products. Some platforms have been outdated; others continue releasing new versions that fix bugs detected or offer new features. A common problem when one wants to benefit from mobile agent platform is the decision about which platform to use. In this paper, we provide up-to-date evaluation of existing mobile agent platforms. We derive linear prediction model to compare and forecast mobile agent platform's performance and their interested behavior. The study uses six mobile agent systems: JAMES, Odyssey, Swarm, Grasshopper, Aglets, and Voyager.

Keywords: Mobile Agents, Performance Analysis, Linear Prediction Model.

1. INTRODUCTION

Performance behavior of mobile agent systems focuses on two main concerns: comparing of agent platforms and platform performance optimization. The most common approaches for analysis performance are developing applications that are concentrated on platform execution time measurement [11]. However, there is no standard analysis for comparing performance optimization. In this paper, a study of mobile agent system analysis is accomplished based on comparing different agent systems that available in the community. Besides, we consider questions that have been arisen in the field of study of agent performance analysis:

- How we compare performance of mobile agent systems?
- What type of experimental benchmarking and environment usage conditions that could be used to experiment agents?
- Is performance the main issue to decide that a given mobile agent system is efficient?
- What type of application area that could be used to judge for the behavior of agents?
- What type of models that suits to make agent performance is applicable?

To address some of these questions, we introduce a certain numerical analysis model that could be used as an introduction in the study of this field of mobile agent optimization; therefore, an evaluation analysis of six agent models has been introduced using a linear prediction model. To best of our knowledge, we report that linear models for approximation can give solutions for forecasting mobile agent system execution time and the measurement of agent system performance.

2. RELATED WORK

Several authors presented a comparative study of mobile agent toolkits. [3][4][5], presented an experimental study that compares performance of eight java-based mobile agent platforms. The study showed that a developed mobile agent platform JAMES is the most robustness and efficient performance over other systems. Implementation of JAMES performance focuses only on 100Kb and 1Mb agent size. [7] [13] conducted a study for comparing the performance of Aglets and TACOMA platforms in distributed search BFS and DFS. Results of comparison indicated that the behavior is quite similar but their performance varies. Performance is better with TACOMA than
Aglets in case of multiple agents in (MANET). [6] presented a methodology for evaluating seven mobile agent systems based on five metrics: availability, environment, development, characteristics, and performance. Results indicated that JADE has the best performance over other platforms. [9] [10] conducted a comparison for performance evaluation of mobile agent systems. The performance evaluation is tested using seven agent systems and required 100 agents which carried out 50 numbers of iterations. Results indicated that the performance for the first three systems gives best results: Springs, Voyager, and Grasshopper, when the number of agents is quite small (2 agents), the performance results regarding the same number of iterations give the optimality as: Aglets, Voyager, and JADE, when number of agents becomes large. [8] presented an analysis and performance of mobile agent systems based on set of features: mechanism of agent mobility, communication scheme, and language support.

These reports only focus on the performance of agent systems according to abstract implementations regarding real scenarios for well known parameters: small agent size, number of iterations, and agent algorithm. Due some limitations in mobile agent systems, the performance of mobile agent systems is undetermined when agent size exceeds 1Mb. However, no performance results have been reported due to agent platform shortcomings. In our research, we focus on studying performance analysis by implementing numerical analysis approaches for the run-time behavior of agent performance using different mobile agent systems with different agent sizes.

3. MOBILE AGENT SYSTEMS

In this section, we present an up-to-date study six agent platforms that could be used in our analysis: JAMES, Odyssey, Swarm, Grasshopper, Aglets, and Voyager. The study presents also a set of features that each platform poses:

**JAMES**: [1] [2] is not a commercial product. Project developed by the University of Coimbra (Portugal) in cooperation with Siemens S.A. for an infrastructure of mobile agents with enhance to support (MANET). The platform is 100% pure Java implementation and the integration with CORBA. JAMES is high performance, secured, robustness platform, and flexible code upgrading. JAMES has been enhanced with comprehensive support for fault tolerance and resource control. Furthermore, JAMES integrates a set of mechanisms to optimize the migration of mobile agents, including cashing, code prefetching, protocol optimization, recycling of threads and sockets.

**Odyssey**: is a Java-based mobile agent system from General Magic. The platform has a transport-independent API that work with RMI, IIOP, and DCOM. Odyssey provides two types of classes-agents and classes which support good functionality for roaming mobile agents involves services for message communication including synchronous messaging scheme. Asynchronous messaging scheme is not supported by Odyssey. This is a drawback to otherwise an excellent platform. More detail can be given in: http://www.genmagic.com/technology/Odyssey.html

**Swarm**: Swarm platform is being developed by Siemens, from the University Stuttgart, Germany. Swarm is being mainly used by the ACTS AMASE Project to provide a middleware for mobile applications in wireless networks. Swarm provides an extensive support and features for inter agent communication scheme.

**Grasshopper**: Grasshopper is a mobile agent platform that has been designed in conformance with MASIF and FIPA standards. It is distributed commercially by Enago Mobile. The platform is implemented in Java. It supports several protocols by the use of an internal ORB, provides GUI to manage agents, agencies, and regions. Furthermore, the platform supports security, agent communication scheme, and agent persistency.

**Aglets**: Java-based system in which agents roam from one-host to another. Aglets is developed by IBM, Tokyo in 1996 and maintained by open source community since 2001. The migration of aglets is based on a proprietary Agent Transfer Protocol (ATP). Agents are controlled by Tahiti Server. Aglets features include: user friendly GUI for programming agents, supports both synchronous and asynchronous messaging scheme, and widely used in the community of distributed computing systems. Static proxies are used by Aglets as a convenient abstraction to refer to remote agents. One disadvantage of this platform is that proxies are not dynamic (i.e. cannot be used after the agents move to another place).

**Voyager**: (http://www.recursionsw.com/) developed by Object Space in 1997 and currently by Recursion Software (last version: Voyager Edge 6.0.1).
Voyager is a platform developed for distributed computing middleware focused towards simplifying the management of remote communications of CORBA and RMI protocols. Voyager has set of features, including: offer facilities such as dynamic generation of CORBA proxies, support for agent communication and agent security, provides flexible life spans for agents, by supporting variety of span methods. Furthermore, Voyager provides location transparency through forwarding chains of proxies, but the whole chain must be traversed to locate agents, otherwise, a single broken link makes the agent unreachable.

4. ENVIRONMENT CONDITIONS

The benchmark application was executed in all the six platforms and results of experiments are given below [12][1]. Results are carried out by changing parameters (number of agencies, number of itinerary performed by the agent, and agent data size). Furthermore, the benchmark application is composed by a single agent with small data size (100 KB, 1 MB). All results were given under six computers (Pentium IV 3.6 GHz, 2 GB RAM, running Windows NT and JAVA standard development Kit with JIT option. The agent is running across the network through a closed itinerary, which is called a "lap" [3]. Table (1) shows results of execution for the six different platforms.

<table>
<thead>
<tr>
<th>Number of Agencies: 1, Lap: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of itinerary performed by the agent: 1</td>
</tr>
</tbody>
</table>

Table 1. Platforms Execution Time Results

<table>
<thead>
<tr>
<th>Platform</th>
<th>Small 100KB</th>
<th>Small 1MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>JAMES</td>
<td>.7</td>
<td>1.17</td>
</tr>
<tr>
<td>Odyssey</td>
<td>.89</td>
<td>1.22</td>
</tr>
<tr>
<td>Swarm</td>
<td>1.01</td>
<td>1.88</td>
</tr>
<tr>
<td>Grasshopper</td>
<td>1.47</td>
<td>2.19</td>
</tr>
<tr>
<td>Aglets</td>
<td>1.73</td>
<td>2.35</td>
</tr>
<tr>
<td>Voyager</td>
<td>1.67</td>
<td>2.37</td>
</tr>
</tbody>
</table>

According to these experiments, JAMES could be considered the platform that offers the best performance while Voyager is the slowest platform. Figure (1) presents platform performance analysis.

5. LINEAR PREDICTION MODEL

The proposed model is derived from Newton first order divided difference formula [14]. Given \( x_0, x_1, x_2, x_3, \ldots, x_n \) \( \exists \ p_n(x_i) \) is a polynomial of degree \( n \) for values \( f(x_i) \) at \( x = x_i \) and \( i = 0, 1, \ldots, n \).

This polynomial approximates a function such that \( p_n(x_i) = f(x_i); \forall i = 0, \ldots, n \)

Let

\[
e_k(x) = \prod_{i=0}^{k-1} (x - x_i), \quad \alpha_k \text{ be a coefficient } 0 \leq k \leq n
\]

for \( f(x_i) = \alpha_i, \forall i = 0, 1, \ldots, n \) (1)

\[
P_n(x_k) = \sum_{k=0}^{n} \alpha_k e_k(x)
\]

(2)

By assuming \( n=1 \), we derived the Linear Prediction Model as follows:

\[
p_n(x_i) = \sum_{k=0}^{n} \alpha_k e_k(x_i) = \alpha_0 + \alpha_1 (x - x_0)
\]

(3)

Using (1) and (2), we get the following

\[
\alpha_i = \frac{f(x_i) - f(x_0)}{x_i - x_0} = f(x_0, x_i)
\]

(4)

\( f(x_0, x_i) \) is called the first order divided difference formula of the Linear Prediction Model (LPM) given by (3).
6. PLATFORMS LINEAR PREDICTION MODEL (LPM)

We use equations given by (3), (4) and Table (1) to compute platform Linear Prediction Models. Table (2), shows the results:

<table>
<thead>
<tr>
<th>PLATFORMRM</th>
<th>$\alpha_0$</th>
<th>$\alpha_1$</th>
<th>LPM</th>
</tr>
</thead>
<tbody>
<tr>
<td>JAMES</td>
<td>0.07</td>
<td>0.00074</td>
<td>0.653 + 0.00047x</td>
</tr>
<tr>
<td>Odyssey</td>
<td>0.89</td>
<td>0.0033</td>
<td>0.857 + 0.00033x</td>
</tr>
<tr>
<td>Swarm</td>
<td>1.01</td>
<td>0.0087</td>
<td>0.923 + 0.00087x</td>
</tr>
<tr>
<td>Grasshopper</td>
<td>1.47</td>
<td>0.0072</td>
<td>1.398 + 0.00072x</td>
</tr>
<tr>
<td>Aglets</td>
<td>1.73</td>
<td>0.0062</td>
<td>1.668 + 0.00062x</td>
</tr>
<tr>
<td>Voyager</td>
<td>1.67</td>
<td>0.0070</td>
<td>1.6 + 0.00070x</td>
</tr>
</tbody>
</table>

Using Table (2), LPM is computed in Table (3), which shows the execution time when agent sizes are very small (10KB), medium (2MB), large (10MB), and very large (1GB) with Lap 1, and number of itinerary performed by the agent is 1.

7. PERFORMANCE RESULTS & ANALYSIS

The platform that offered the best performance in our previous results when agent data size is very small is JAMES platform, which is the efficient platform (0.6577 seconds), while Aglets platform offered the slowest execution time (1.6742). Figure (2), shows results execution time. Thus, Platforms are ordered (faster →slower) according to platform execution time as:

(JAMES →Odyssey→Swarm→Grasshopper→Voyager→Aglets)

![Figure 2. Platform Performance Comparison with (10 KB) Very Small Agent Data Size](image-url)
In case when agent has medium data size (2 MB), the platform that offered best execution time is Odyssey 1.517 seconds), while Voyager offered the slowest platform (3.000 seconds. Figure (3) shows results. Thus, platforms are ordered (faster→slower) according to platform execution time as:
(Odyssey→JAMES→Swarm→Grasshopper→aglets→Voyager)

![Figure 3. Platform Performance Comparison with (2 MB) Medium Agent Data Size](image)

In case when agent has large data size (10 MB), the platform that offered best execution time is Odyssey 4.157 seconds), JAMES still offering the second fastest platform with (5.353 seconds) execution time, Aglets transferred to the third fastest platform with (7.868 seconds) execution time, while Swarm offered the slowest platform (9.623 seconds). Figure (4) shows results. Thus, platforms are ordered (faster→slower) according to platform execution time as:
(Odyssey→JAMES→Aglets→Grasshopper→Voyager→Swarm)

![Figure 4. Platform Performance Comparison with (10 MB) Large Agent Data Size](image)

In case when agent has very large data size (1 GB), the platform that offered best execution time is (Odyssey 330.857 seconds). Swarm offered the slowest platform with (870.923 seconds). Order of platforms is quite similar to that with large agent data size except that Voyager platform offered best execution time (701.6 seconds) than Grasshopper (721.398 seconds). Figure (5), shows results. Thus, platforms are ordered (faster→slower) according to platform execution time as:
(Odyssey→JAMES→Aglets→Voyager→Grasshopper→Swarm)

![Figure 5. Platform Performance Comparison with (1 GB) Very Large Agent Data Size](image)

8. CONCLUDING REMARKS AND FUTURE DIRECTIONS

In this research paper, we survey six mobile agent systems with analysis of performance characteristic. We developed a mathematical Linear Prediction Model (LPM) that forecasts platforms performance behavior using their execution time with one agent and different agent data size including (very small 10KB, Medium 2MB, Large 10 Mb, and very large 1 GB). The performance analysis using LPM showed that the Odyssey offered the best performance when agent data size becomes medium, large, and very large, while Swarm offered the slowest performance when agent data size is large and very large. The performance of JAMES is efficient when agent data size is very small and small. Aglets offered a quite good performance comparing with other agent platforms when agent data size becomes large and very large. The other platforms including Voyager and Grasshopper do not offer a good performance than Odyssey, JAMES, and Aglets for large and very large data size, but they offer a good performance comparing to Swarm.

Future directions will include study of mobile agent systems with two main categories:
1. Analysis of the six mobile agents performance in case with multiple agents using LPM.

2. Expect to contribute and cover the need for other mobile agent systems for analysis their performance behavior including: Tryllian, JADE, Tracy, SPRINGS, TACOMA, Concordia, and Jumping Beans.

We believe that all platforms are interested, and that our comparison could help to developers to decide which platform suites his/her needs. With this piece of work, we could encourage future work in the field of mobile agent's performance analysis.

REFERENCES:


