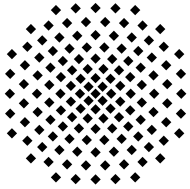

Universität Stuttgart
Fakultät Informatik

Efficient Algorithms to Find Optimal Agent Migration Strategies

Authors:
Prof. M. Ashraf Iqbal
Dipl.-Inform. Joachim Baumann
Dipl.-Inform. Markus Straßer

Institut für Parallele und Verteilte
Höchstleistungsrechner (IPVR)
Fakultät Informatik
Universität Stuttgart
Breitwiesenstr. 20-22
D-70565 Stuttgart
Germany

Efficient Algorithms to Find Optimal Agent Migration Strategies

A. Iqbal, J. Baumann, M. Straßer

Bericht 1998/05
April 1998

Efficient Algorithms to Find Optimal Agent Migration Strategies

M. Ashraf Iqbal* Joachim Baumann
Markus Strasser

Institute for Parallel and Distributed High-Performance Systems (IPVR)
University of Stuttgart, Breitwiesenstr. 20-22, D-70565 Stuttgart, Germany

Abstract

Mobile agent technology has received a rapidly growing attention over the last few years. A number of mobile agent systems are under development in academic as well as industrial environments, there are already various moves to popularize and standardize mobile agent facilities and architectures. It has been argued that mobile agents provide certain advantages as compared to traditional approaches in terms of providing facilities like software-distribution on demand at reduced communication costs, better support for asynchronous tasks, and scalibility due to dynamic deployment which is based on a taxonomy of mobility. The employment of mobile agents has been particularly attractive in applications like information retrieval in widely distributed heterogeneous environments, network management, electronic commerce and in mobile computing.

Inspite of all this growing interest and efforts the agent technology is in a fairly early stage and a number of technical problems should be solved in order to make this new technology a commercial success. These open problems include mechenisms for agent security, control structures, transactional support, and design of communication models. In this paper we extend previous research conducted on mobile agents regarding their communication performance models. It has been indicated that the optimal performance of an agent is achieved by a critical sequence of mixed remote procedure calls and agent migration. We provide here exact algorithms to solve various variants of this problem under certain restrictions on the sequence of interactions of the mobile agents.

*Currently with the Department of Electrical Engineering, University of Engineering & Technology, Lahore-54890, Pakistan, email:ashraf@iqbal.lhr.aster.com.pk.

1 Introduction

Mobile agents are computer programs which can be migrated from a client computer and dispatched to a remote host for execution [BRAD97]. The **Mobile Agent technology**, which implements the concept of mobile agents in a commercial environment, is a new promising technology moving towards the vision of usuable distributed systems in distributed heterogeneous open networks [BAUM98,LANGE98,PEIN97, POPE97]. This promises to offer an appropriate framework for a unified and scalable electronic market. The salient features of this new technology is that a user computer and a server can interact without using the network once the network has transported an agent between them. Thus any ongoing interaction does not need ongoing communication between a user and a server computer [BRAD97].

The current organizing principle of computer communication networks is **Remote Procedure Calling (RPC)**. Designed in the seventies, the RPC paradigm models computer to computer communication as enabling one computer to call procedures in the other machine. Each message that is sent on the network transports either requests (i.e., procedure's arguments) or acknowledges a procedure's performance in terms of data which are its results. The salient features of RPC is that each interaction between the user computer and the server consists of two steps of communication, in the first step the server is requested to perform an operation on a given set of arguments, and in the second step is to acknowledge or receive the results of the operation from the server. Thus under RPC ongoing interaction requires ongoing communication.

Under this new technology one computer not only calls procedures in another machine but also supplies procedures to the other machine for execution. A client computer, with functions to be performed by a server, sends to the server an agent while the agent and not the client computer orchestrates the work, making on site decisions while performing its job. Thus under agent technology ongoing interaction does not require ongoing communication. The public telephone systems and today's wireless networks presents greater opportunities for the agent technology as under such systems a user computer is connected to a network occasionally rather than permanently. The client machine should be connected to the network just long enough to send the agent on its way, and later to receive it at home, it needs not be connected while the agent carries out its assignment.

Basic Concepts

The commercial focus of mobile agent technology is the **electronic marketplace** consisting of a public network that will design a platform for providers and consumers

of goods and services where business transactions are carried out electronically. The technology of the so called future marketplace is based on the following concepts. The network of computers is modelled as a collection of **places**, each place provides a service and the agent can migrate to any place to request for the service it provides. An agent occupies a particular place, it can move from place to place thus occupying different places at different times. **Travel** from one place to another provides a facility for the agent to obtain a service offered remotely and is the most distinguished feature of this technology. Two different agents can **meet** in the same place, a meeting lets agents in the same computer to call one another's procedures. An agent's **travel** is not restricted to a migration to a single place, the agent is powerful or independent enough to travel to several places in succession, taking advantage of the basic services of the places it visits, the roaming agent can thus provide a sophisticated range of composite services.

Several researchers have suggested that mobile agents provide an innovative technique of performing transactions and information retrival in computer networks [HARI95]. Other researchers have, however, indicated a number of serious problems which must be overcome before the agent technology becomes commercially useful and cost effective [ROTH97]. The following are some of the problems which should be tackled by researchers working in the said field.

- **Virus scanning** and **epidemic control** mechenisms should be designed and performance limitations resulting from security issues should be well understood. In other words there is a growing need for highly secure agent execution environments.

- The problem of migrating agent execution environments onto a large number of third party servers should be solved. In particular the third party servers should appreciate the potential advantage of allowing or **supporting the computational load of mobile agents**.

- Due to their asynchronous nature, agents are best suited to be engaged in long-lived activities most probably where it has to migrate to a large number of places in a sequence in order to provide a number of composite services. A communication model should be developed which can help us to **design an optimal agent migration strategy** exactly specifying places where an RPC is cost effective and others where the agent should migrate in order to provide the required services at minimal cost.

In this paper we extend previous research conducted on mobile agents regarding their communication performance models. It has been indicated that the optimal

performance of an agent is achieved by a critical sequence of mixed remote procedure calls and agent migration. We provide exact algorithms as well as approximate schemes to solve this problem provided the sequence of interactions is fixed and can be represented by a constrained graph. We also study the performance model when the sequence of interactions is not fixed, we provide approximate solutions and fast heuristics for this otherwise NP-complete problem. The paper is organized in the following manner. In Section 2 we describe an exact algorithm for finding an optimal agent migration strategy provided the sequence of interactions is fixed and is represented by a chain structure. We extend this algorithm for tree structured and series-parallel graphs in Section 3. We conclude the paper in Section 4.

2 Agent Migration versus RPC

The key advantage of agent migration as compared to RPC lies in the reduction of expensive global communication costs and that is achived by moving the computation to the place which houses the relevant information or services. Under what conditions this argument is true or false? In order to answer this question one has to evaluate the performance of agent migration as compared to communication by RPC on a quantitative fashion. A simple performance model was studied in [CARZ97], [CHIA97]. Another performance model for mobile agent systems where agents can alternatively use RPC or migrate to different places is considered in [MARK97]. It has been assumed that the sequence of interaction partners and their locations, as well as each request size, reply size, selectivity, and number of communications per location is known in advance. It has also been assumed that the average delays and the average throughput in the network are already known for every possible interconnection.

2.1 Assumptions

It has been shown [MARK97] that a particular alternating sequence of of agent migrations and remote procedure calls performs better than a simple sequence of RPC's or a sequence of agent migrations to all possible places. The theoretical result was confirmed by actual measurements on a prototype implementation of a mobile agent system known as the Mole. In this section we work under the same constraints as are made in [MARK97], we thus assume that the complete sequence of interactions with different places is fixed. We also assume that the cost of interaction under RPC as well as under agent migration is also known before hand.

2.2 Definitions

The agent communicates with a number of places in a fixed sequence. The set of places with which the agent communicates is denoted by **SCP**, the size of this set is denoted by y. The agent is allowed to migrate to a specified number of locations, these locations are represented by the set **SMP**, the size of this set is assumed to be equal to z. We assume that the agent communicates with places (belonging to the set SCP) in a fixed sequence, the sequence is known as **CS**, we assume that the communication takes place n times. Thus $CS(i)$ specifies the place with which the ith communication takes place. In order to communicate with different places the agent migrates, the migration sequence is represented by **MS**. Thus $MS(i)$ specifies the position of the agent when the ith communication takes place, $1 \leq i \leq n$. It is important to note that a single communication step is composed of two basic operations:

- In the first operation the **agent migrates** from $MS(i-1)$ to its new location which is $MS(i)$.
- In the second operation the agent makes a **remote procedure call** to location $CS(i)$.

Let us assume that **MCost(a,b)** is the cost of migrating the agent from location a to location b. We represent the cost of making a remote procedure call from location c to location d by **RCost(c,d)**. Thus the **Cost** of making an ith communication would be equal to the sum of the above two costs and is given by the following equation.

$$Cost(i) = MCost(MS(i-1), MS(i)) + RCost(MS(i), CS(i))$$

The total cost of communication would be $\sum_{i=1}^{n} Cost(i)$, and is donated by **TCost**.

Example 1

Let us assume that the agent communicates 5 times, thus n=5. Further assume that $SCP = \{1, 2, 3\}$, and $SMP = \{1, 2, a, b, c, d\}$. The communication sequence is given by the following equation.

$$CS = \{2 \rightarrow 3 \rightarrow 1 \rightarrow 2 \rightarrow 3\}$$

We assume that the agent is currently residing at place 1 and the migration sequence is given below.

$$MS = \{b \rightarrow c \rightarrow 2 \rightarrow 2 \rightarrow b\}$$

Thus in order to communicate for the first time, i.e., with $CS(1) = 2$, the agent migrates to location $MS(1) = b$ from its present position, which is 1, and then makes a remote procedure call to location 2. Similarly $CS(4) = 2$, while $MS(4) = 2$, and the old position of the agent is $MS(3) = 2$, that means that the agent will stay at location 2 and shall make a local communication with location 2. The last communication takes place with $CS(5) = 3$, the agent migrates to location b from its old position and then makes a remote procedure call to location 3, after that the agent returns to its home position which was location 1. The value for $Cost(i)$ would be given by the following equations:

$$Cost(1) = MCost(1, b) + RCost(b, 2)$$
$$Cost(2) = MCost(b, c) + RCost(c, 3)$$
$$Cost(3) = MCost(c, 2) + RCost(2, 1)$$
$$Cost(4) = MCost(2, 2) + RCost(2, 2)$$
$$Cost(5) = MCost(2, b) + RCost(b, 3)$$

The total cost would be the sum of all the above costs, we assume here that the agent does not return to its starting position which is location 1. If it does return to its home position then an extra cost equal to $MCost(b, 1)$ should be added into the above costs.

Problem Definition

Given a fixed communication sequence in terms of $CS(i)$, $1 \leq i \leq n$, the migration cost matrix $MCost(a, b)$, the remote procedure call cost matrix $RCost(c, d)$, find a migration sequence MS so that the total cost of communication $TCost$ is minimal. Note that MCost is a $z \times z$ matrix while RCost is a $z \times y$ matrix

2.3 An Optimal Decision Graph (ODG)

We can find an optimal agent migration sequence, i.e., the migration sequence of minimal total cost by drawing an optimal decision graph, this is a layered graph having as many layers as the size of the communication sequence, i.e., n. In addition to the n layers the ODG has a start and an end node. The start node signifies the starting or the home location of the agent while the end vertex corresponds to the final destination of the agent. A shortest path between the start and the end node in this graph determines the optimal migration sequence [ASHR96, GABR97]. The optimal decision graph has the following properties:

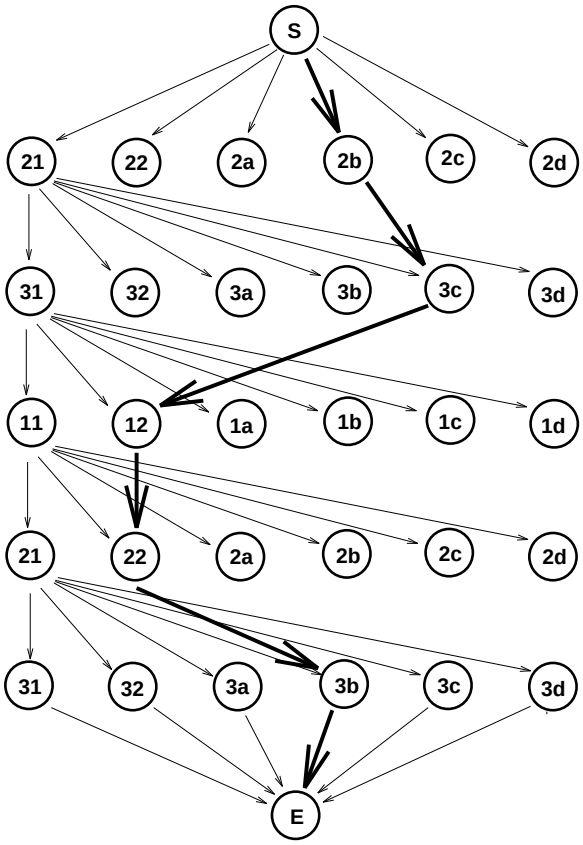

Figure 1: An Optimal Decision Graph (ODG) indicating a path between the start and the end node. Each such path represents an agent migration sequence. In order to avoid confusion all edges are not shown in the graph.

- There are n layers in the ODG, the ith layer corresponds to the ith communication step in the communication sequence $CS(i)$, $1 \leq i \leq n$.
- Each layer i has as many vertices as the size of SMP, i.e., z, we label these vertices as ab, where $a = CS(i)$, and $b \in SMP$.
- A path passing through vertex ab in the ith layer, where $a = CS(i)$, and $b \in SMP$, signifies that the ith communication step takes place while the agent

is currently residing at location b and from that location it makes a RPC to location a.

- The start node is connected to each vertex ab in the first layer, where $a = CS(1)$ and $b \in SMP$. This implies that the agent has migrated from its home location to location b and makes an RPC from b to a. The cost associated with edge$(start, ab)$ is $Cost(1) = MCost(home, b) + RCost(b, a)$.

- Each vertex ab in layer $i-1$ is connected to each vertex cd in layer i, the cost associated with this edge is $Cost(i) = MCost(b, d) + RCost(d, c)$. There are z incoming edges at each vertex cd in layer i from each vertex in layer $i-1$.

- Each vertex ab belonging to the nth layer is connected to the end node, each such edge signifies that the agent returns from its last position (which is b) to its home vertex, note that $a = CS(n)$. The cost associated with edge(ab, end) is $MCost(b, home)$.

- There is a one to one correspondence between an agent migration sequence and a path between the start and the end node in the optimal decision graph. The cost of the agent migration sequence is represented by the length of the corresponding path in the optimal decision graph. In order to find the optimal agent migration sequence we should find a shortest path between the start and the end vertex in the optimal decision graph.

Example 2

An optimal decision graph corresponding the agent migration problem described in Example 1 is shown in Fig. 1. The following information is reproduced from Example 1.

$$SCP = \{1, 2, 3\}$$
$$SMP = \{1, 2, a, b, c, d\}$$
$$CS = \{2 \rightarrow 3 \rightarrow 1 \rightarrow 2 \rightarrow 3\}$$

We assume that the agent is currently residing at place 1 and the migration sequence is given below.

$$MS = \{b \rightarrow c \rightarrow 2 \rightarrow 2 \rightarrow b\}$$

Note that any agent migration sequence can be represented by a distinct path between the start and the end nodes in the optimal decision graph. Consider the MS, given above, the corresponding path in the optimal decision graph is shown in bold in Fig. 1. The length of this path is equal to the cost of the MS.

An Algorithm to Find an Optimal Agent Migration Sequence

The problem is to find a shortest path in the optimal decision graph. We consider all outgoing paths from each vertex ab belonging to the second last layer, i.e., the $n-1$ layer.

1. Find a shortest path from each vertex ab to the end node. This would involve a number of comparisons equal to the square of the size of SMP, i.e., z^2.

2. Move one layer up at a time until you reach the start vertex, and consider each vertex ab in that layer, now go to step number 1. Processing of each layer will take time proportional to z^2.

The total complexity of the algorithm would be $O(nz^2)$.

3 When Comunication Sequence is Not Fixed

It has been assumed in the last section that the agent communicates with different places in a fixed sequence, i.e., there is only a single place with which the ith communication can take place, where $1 \leq i \leq n$. The communication sequence can thus be represented by a chain structured graph, there are n vertices in this graph connected in the form of a chain, the ith vertex represents the ith communication step. For example, if $CS = \{2 \rightarrow 3 \rightarrow 1 \rightarrow 2 \rightarrow 3\}$, and $MS = \{b \rightarrow c \rightarrow 2 \rightarrow 2 \rightarrow b\}$, as in Example 1, then this information can be represented by a chain structured graph as shown in Fig. 2. For each vertex i in this graph, $CS(i)$ is shown at the top of vertex i while $MS(i)$ is indicated in the bottom of each vertex. Note that in a graph, representing a fixed communication sequence, the out degree of each vertex is not more than one. In this section we consider cases where the communication sequence is not fixed and is represented by tree or series-parallel graphs.

3.1 When CS is Tree Sructured

A tree structured communication sequence graph is shown in Fig. 3. Note that after communication step 2, the agent can communicate with place 1 or with place 2, the communication sequence is thus not fixed, and the out degree of a vertex in a graph, representing such a situation, can be more than 1. We handle this situation by cloning the agent, i.e., the original agent can take care of communication steps 3, 4, and 5 (Fig. 3), while the clone, known as agent* is responsible for communication steps 3*, 4*, and 5*. A possible migration sequence $MS(i)$ is also indicated at the bottom of each vertex i in the graph of Fig. 3.

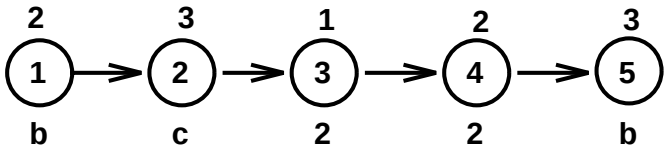

Figure 2: A chain structured graph represents a fixed communication sequence.

The Migration Tree

The vertices of the optimal decision graph for a tree structured communication sequence graph are shown in Fig. 4. The ODG is so designed that each migration sequence corresponds to some subset of vertices of the ODG. The subgraph generated by these vertices in the ODG is a tree, known as the **Migration Tree**. It is important to note that the migration tree contains one and only one node from each layer of the ODG, and there is a one to one correspondence between a migration tree and the migration sequence. The migration tree within the ODG, shown in Fig. 4, corresponds to the migration sequence of Fig. 3.

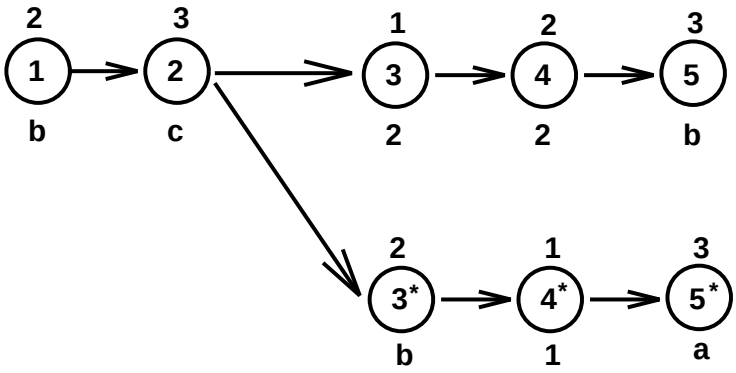

Figure 3: A tree structured communication sequence graph.

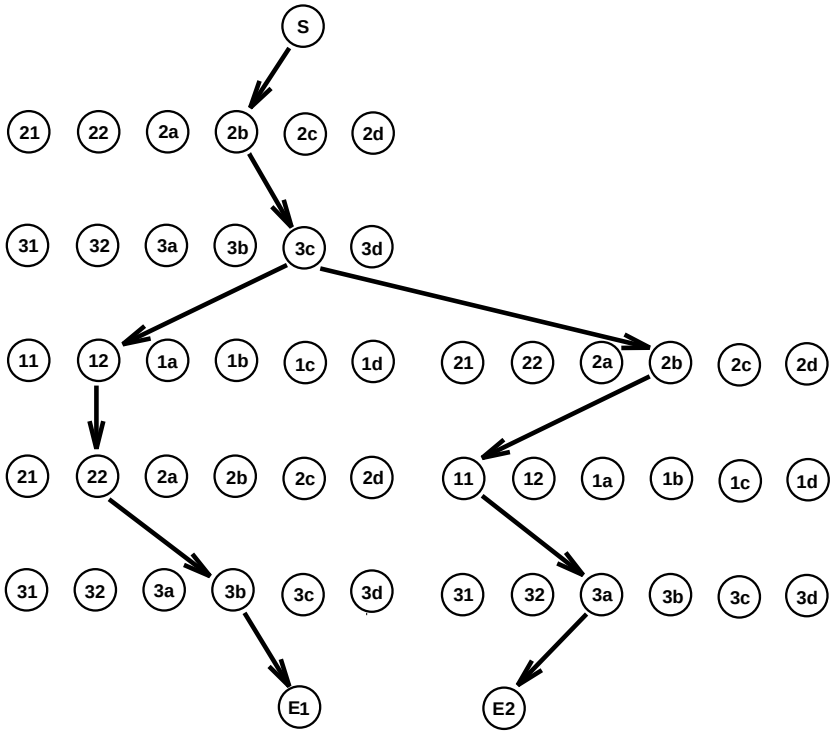

Figure 4: An Optimal Decision Graph for a tree structured communication sequence graph. A migration tree is also shown within the ODG. Note that all edges of the ODG are not indicated.

Finding the Optimal Migration Tree

Consider the tree structured communication sequence graph as shown in Fig. 3. Remember that vertex 2 of this graph is a so called **Fork** vertex, and it is the presence of fork vertices in the CS graph which differentiates a tree(Fig. 3) from a chain structure(Fig. 2). Corresponding to this fork vertex in the CS graph, there are a total of z fork vertices in the ODG as shown in Fig. 4. From each such vertex there are two outgoing paths, one leading towards $E1$, while the other path terminating at $E2$. We shall find the shortest path from each such fork vertex to each of the end vertices. For example let us assume that the shortest path from vertex $3c$ to E1 is $3c \rightarrow 12 \rightarrow 22 \rightarrow 3b \rightarrow E1$, and from $3c$ to $E2$ it is $3c \rightarrow 2b \rightarrow 11 \rightarrow 3a \rightarrow E2$.

We represent these two paths by a single edge from vertex $3c$ to a pseudo end node represented by E, the weight of this edge is equal to the sum of the two paths it represents as shown in Fig. 5. We use this procedure in a **bottom up** fashion to find the optimal migration tree as described in the following algorithm.

The Algorithm

We know that there is a one-to-one correspondence between a migration tree and a migration sequence, and that the weight of each migration tree, i.e., the sum of the weights of all edges in it, equals the cost of the migration sequence. Thus in order to find the optimal migration sequence we should find the migration tree of minimum weight. Corresponding to each fork vertex in the communication sequence graph, there are z fork vertices in the ODG. From each such fork vertex xy find shortest paths from xy to each of the end vertices. Merge these shortest paths into a single edge between vertex xy and a pseudo end vertex in the simplified ODG. Move in the ODG in a **bottom up** fashion unitil the minimal weight migration tree is found. The complexity of the algorithm would still be $O(nz^2)$.

3.2 When CS is a Series-Parallel Graph

In the last section it was assumed that the communication sequence can be represented by a tree structured graph like the one shown in Fig. 3. Under such conditions the original agent is responsible for communication steps 1, 2, 3, 4, and 5, while the clone handles communication steps 3*, 4*, and 5*. The agent after communication step 5 may return to its home location while the clone after communication step 5* may return to a different home location. It is very much possible that the clone may be assigned a relatively small number of communication steps and after that it reports its findings to the original agent and terminates itself. Under such a scenario the communication sequence would no longer be represented by a tree but would be expressed by a series-parallel graph as shown in Fig. 6. We assume that:

$$SCP = \{1, 2, 3, 6\}$$

$$SMP = \{1, 2, a, b, c, d\}$$

For each vertex i in the graph of Fig. 6, $CS(i)$ is indicated at the top of each vertex i, while $MS(i)$ is shown at the bottom of each vertex. The agent is responsible for communication steps 1, 2, 3, 4, & 5. It migrates to location b and makes an RPC to location 2 in the first communication step. From location b, it migrates

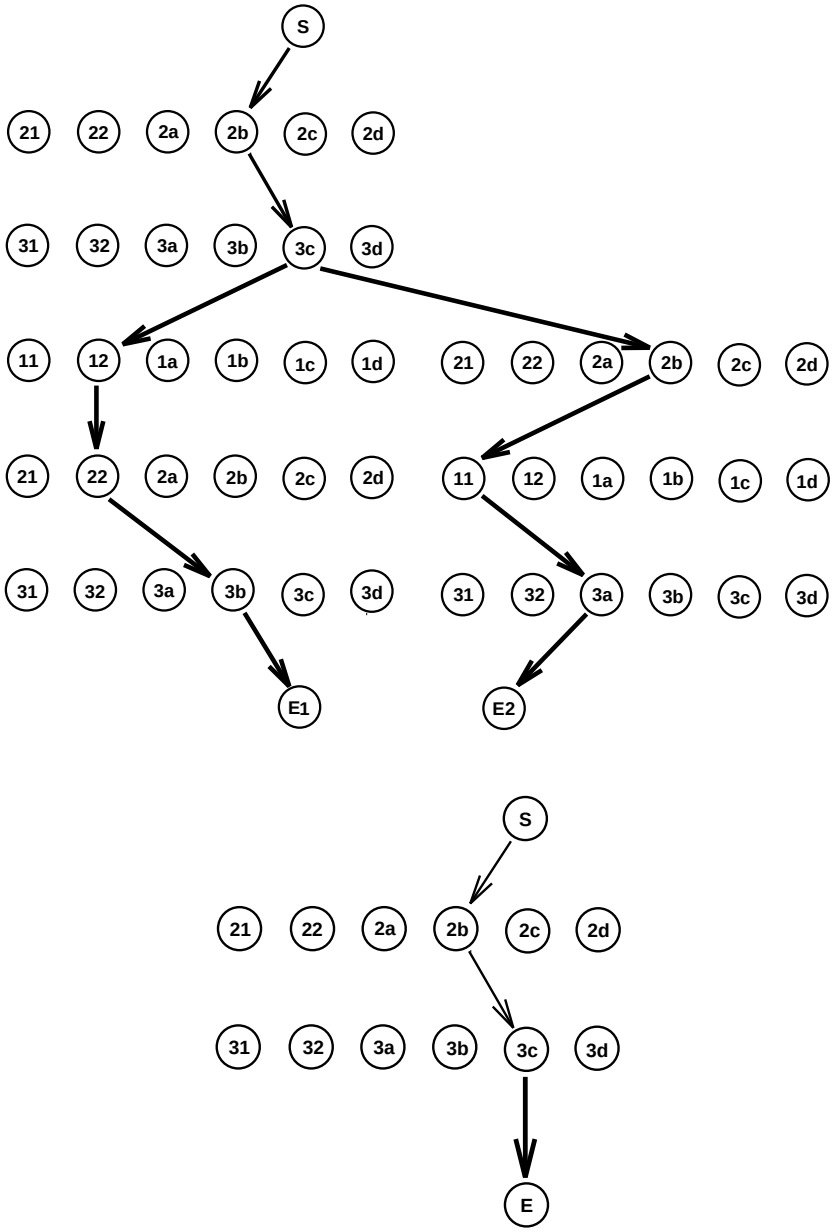

Figure 5: An Optimal Decision Graph with two paths emerging from vertex $3c$ shown in bold (top). The two paths are merged into a single bold edge in the simplified ODG shown in the bottom.

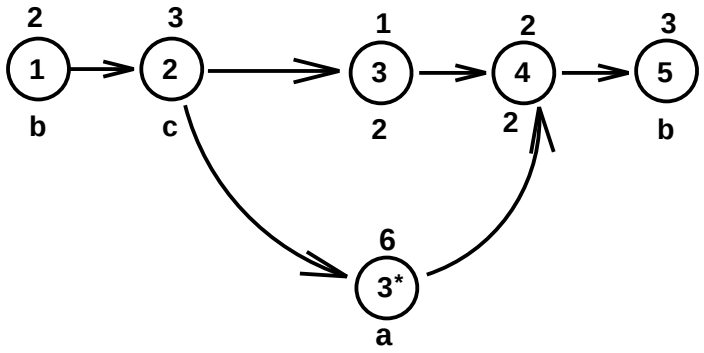

Figure 6: A communication sequence in the form of a series-parallel graph.

to location c, and makes an RPC to location 3 in the second communication step. After this communication step, the agent migrates to location 2 and stays there till communication step 5. The clone activates itself during communication step 2, it migrates to location a, makes an RPC to location 6, and reports its findings to the agent who is residing at location 2. The clone, after finishing its task, terminates itself, and the rest of the communication steps are taken care off by the original agent alone.

The Optimal Decision Graph

An optimal decision graph for such a situation is shown in Fig. 7. The path $s \rightarrow 2b \rightarrow 3c \rightarrow 12 \rightarrow 22 \rightarrow 3b \rightarrow E$ infact represents the migration sequence for the agent while the path $3c \rightarrow 6a \rightarrow 22$ represents the migration sequence for the clone. Note that there are two **Parallel Paths** emerging from vertex $3c$, and terminating at vertex 22 as shown in bold in Fig. 8(top). Path P1= $3c \rightarrow 12 \rightarrow 22$ represents the migration sequence of the agent, while P2=$3c \rightarrow 6a \rightarrow 22$ represents the migration of the clone. The two parallel paths can be merged together into a single edge in the simplified optimal decision graph as shown in Fig. 8 (bottom). The weight of the edge$(3c, 22)$ is equal to the sum of the two parallel paths. This transformation of **multiple parallel paths** into a **single edge** provides a very powerful technique which helps us find the optimal migration sequence for a communication sequence which can be represented by a series-parallel graph.

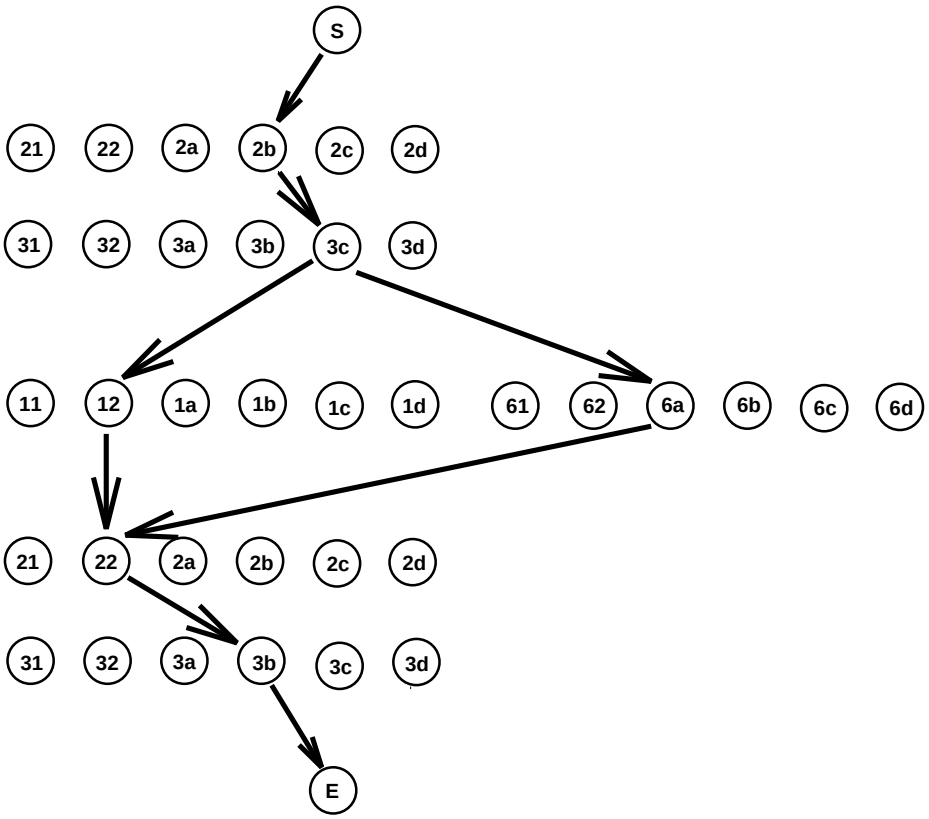

Figure 7: An Optimal Decision Graph (ODG) for a series-parallel communication sequence graph.

The Algorithm

The Optimal Decision Graph is so constructed that the subgraph induced by selecting one vertex from each layer corresponds to a migration sequence. One such subgraph, shown in Fig. 7, represents the migration sequence shown in Fig. 6. The weight of the subgraph is equal to the cost of the migration sequence, thus in order to find the optimal migration sequence, we should find the subgraph of minimal weight.

Find two layers in the ODG which are connected by parallel paths. Find shortest paths from each node in the top layer to each node in the bottom layer. These paths can be combined to get z^2 sets of shortest path combinations, one for each top and

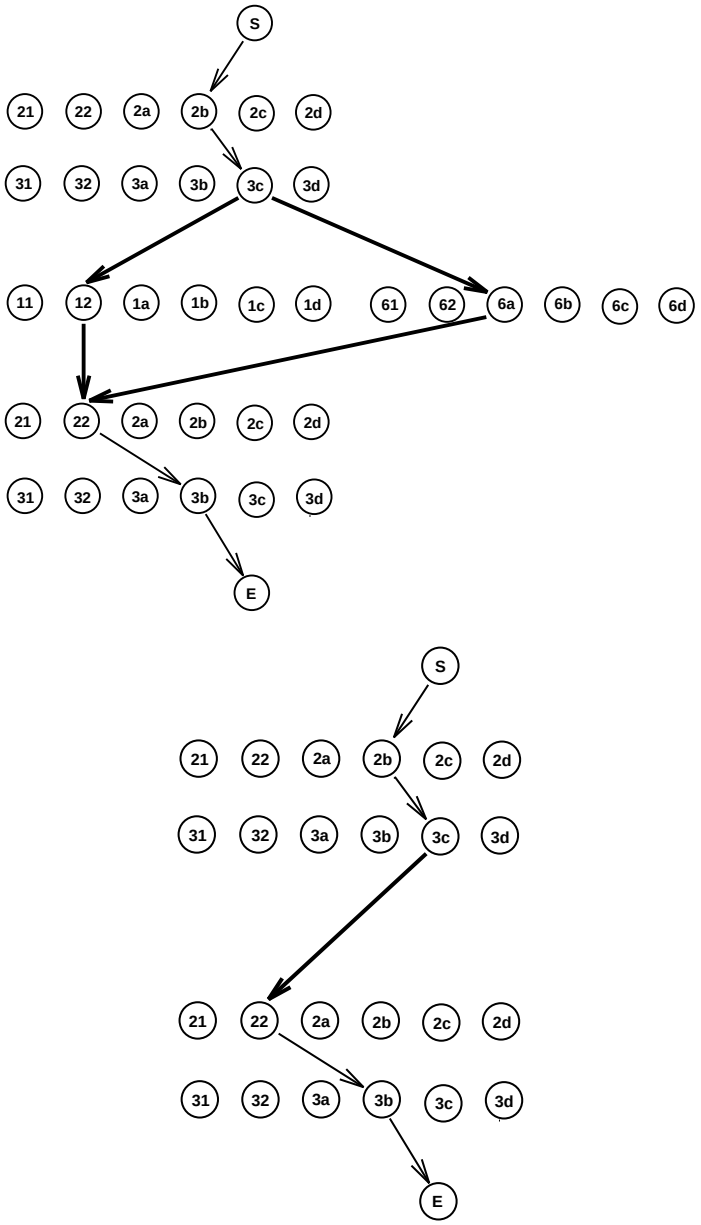

Figure 8: An Optimal Decision Graph (ODG) with two paths emerging from vertex $3c$ and terminating at vertex 22. The two paths, shown in bold(top), are merged into a single bold edge in the simplified ODG shown in the bottom.

bottom vertices, with total weight equal to the sum of the individual paths. The parallel paths are thus merged into a single edge in the simplified ODG. Repeat this process until all parallel paths are transformed into a single edge. The complexity of the resulting algorithm would be $O(nz^3)$.

4 Conclusions

Mobile agent technology has recently received a rapidly growing attention over the last few years. Development of mobile agent systems is under way in both academic and industrial environments. In addition, there are already various efforts to standardize mobile agent facilities and architectures. There are, however, a number of technical problems that have still to be solved to allow the exploitation of agent technology on a large scale. One of these problems is the support for a communication performance model, that allows agents to decide if, given a communication pattern, they should migrate to a site and communicate locally, or the communication should be done remotely. In this paper we have extended previous research conducted on mobile agents regarding their communication performance models. It has been indicated that the optimal performance of an agent is achieved by a critical sequence of mixed remote procedure calls and agent migration. We have provided here exact algorithms to solve various variants of this problem under certain restrictions on the sequence of interactions of the mobile agents.

References

[**ASHR96**] M. Ashraf Iqbal & Alexander Hagin, Partitioning and Mapping Techniques for Distributed Multimedia Applications, Technical Report 14/1996, Institute for Parallel and Distributed High-Performance Systems (IPVR), Stuttgart, Germany.

[**BAUM98**] Baumann, Hohl, Rothermel, Strasser, Mole - Concepts of a Mobile Agent System, accepted for WWW Journal, Special issue on Applications and Techniques of Web Agents, Baltzer Science Publishers, 1998.

[**BAUM97**] J. Baumann, F. Hohl, N. Radounikles, K. Rothermel, & M. Strasser, Communication Concepts for Mobile Agent Systems, Proc. 1st Int. Workshop on Mobile Agents MA97, Springer Verlag, 1997.

[**BOKH87**] S. H. Bokhari, Assignment Problems in parallel and distributed computing, Kluwer Academic Publishers, 1987.

[**BRAD97**] M. Bradshaw, Software Agents, Menlo Park, CA: The MIT Press 1997.

[**CARZ97**] A. Carzaniga, G.P. Picco, & G. Vigna, Designing Distributed Applications with Mobile Code Paradigms, Proc. 19th. Int. Conf. on Software Engineering, Boston, 1997.

[**CHIA97**] T.H. Chia & S. Kannapan, Strategically Mobile Agents, Proc. 1st Int. Workshop on Mobile Agents MA97, Springer Verlag, 1997.

[**GABR97**] Gabriel Dermler & M. Ashraf Iqbal, Task Allocation in Distributed Multimedia Systems based on the Host-Satellite Model, Technical Report, Institute for Parallel and Distributed High-Performance Systems (IPVR), Stuttgart, Germany.

[**HARI95**] C.G. Harrison, D.M. Chess, & A.Kershenbaum, Mobile Agents: Are they good idea?, Research Report, IBM Research Division, T.J. Watson Research Center, Yorktown Heights, NY 10598.

[**LANGE98**] Lange & Oshima, Mobile Agents with Java: The Aglet API, Special issue on Distributed World Wide Web Processing: Applications and Techniques of Web Agents, Baltzer Science Publishers, 1998.

[**MARK97**] Markus Strasser & Markus Schwehm, A Performance Model for Mobile Agent Systems, Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications PDPTA'97, 1997.

[**PEIN97**] Peine, H. & Stolpmann, T., The architecture of the Ara platform for mobile agents, Proc. 1st Int. Workshop, MA'97, Springer, 1997.

[**POPE97**] Rothermel, K., & Popescu-Zeletin, R., Mobile Agents, First International Workshop MA '97, Computer Science, Vol. 1219, Springer, 1997, pp. 86-97.

[**ROTH97**] K. Rothermel, F. Hohl, & N. Radouniklis, Mobile Agents Systems: What is Missing? Invited Paper to the IFIP WG 6.1 International Working Conference on Distributed Applications and Interoperable Systems (DAIS'97).