Network Management Itineraries for Mobile Agents

Damianos Gavalas

Department of Product & Systems Design Engineering,
University of Aegean,
Hermoupolis, Syros, Greece
E-mail: dgavalas@aegean.gr

Abstract - Proposed distributed management architectures, incorporating mobile agent (MA) technology, have not adequately management scalability problems, when monitoring tasks requiring the employment of itinerant agents is considered. Mechanisms building optimal agents itinerary are needed to minimise the total migration cost in terms of the round-trip latency and the incurred traffic. This is of particular importance when the management of networks spanning multiple subnets is involved. On this direction, we have designed and implemented an algorithm which adapts methods usually applied for solving network design problems in the specific problematic of MA itinerary planning. The algorithm not only suggests the optimal number of MAs that minimise the overall cost but also constructs optimal itineraries for each of them. The algorithm implementation has been integrated into our MA framework research prototype and tested into real network environments, demonstrating significant management cost savings.

1. INTRODUCTION

The increasing complexity of networks has motivated the evolution of existing management models. These models traditionally adopt a centralised, Client/Server (CS) approach wherein the functionality of management entities is rigidly defined at design time. Concerning formal standardisation approaches in IP Network & Systems Management (NSM), the scene has been dominated, during the past decade, by the IETF Simple Network Management Protocol (SNMP)[13]. Despite its wide deployment base, SNMP follows the CS paradigm, typically associated with massive transfers of management data, which cause considerable strain on network throughput and processing bottlenecks at the manager host. The situation seriously deteriorates when considering managed networks that span multiple subnets or, even worse, include remote subnets connected to the manager site through expensive, low-bandwidth links. In such cases, the traffic associated with management tasks typically traverses several network segments and, when summed up, results in increased bandwidth waste [12].

These problems have motivated a trend towards distributed management intelligence that represents a rational approach to overcome the limitations of centralised NSM. The new trend in NSM involves using mobile agents (MAs) [3] to manage distributed network systems [2][8][9][10]. An MA can be used to locally retrieve and filter management data to monitor systems health and networking conditions in distributed environments. In particular, management tasks are assigned to an agent which delegates and executes management logic in a distributed and autonomous fashion. After completing these tasks, the results are either communicated through a messaging mechanism or carried back to the manager by the MA.

Delegation of management logic may be realized with agents bound to *single-hop* mobility; the agents move from the managing node to remote managed nodes, where they statically execute their tasks [1]. What is not commonly exploited in management is the MA *multiple-hop* capability, where agents may move several times as they adapt to changing circumstances. While single-hop mobility can improve flexibility and scalability in the context of relatively static networked systems, it is the multiple-hop capability offered by MAs that needs to be exploited to meet the requirements of future networked systems, i.e. large scale and dynamics [8]. In addition, network monitoring based on multi-hop (itinerant) MAs is also advantageous for the following reasons (for a complete discussion of these issues, the interested reader may refer to [4]):

- *Short-term distributed tasks*: When distributed tasks are intended to run over a set of devices for a relatively short period, it is more efficient to use a multi-hop MA to sequentially visit the devices rather than broadcasting mobile code and obtain the results from every network element (NE). That also reduces the code deployment time.
- *Local vs. global semantic compression of data*: Single-hop mobility only involves a level of data filtering limited to a single device, and local to it. The itinerant MA paradigm, in contrast, enables global semantic filtering of data across all the network devices visited [9].

- *Device vs. Domain-level view*: In parallel to the previous argument, approaches based on single-hop mobility imply a local view of the device where the distributed code statically resides. Itinerant MAs offer a more simple solution to this problem as they can perform data correlation through sequentially visiting the entire set of devices.

However, while in single-hop mobility, agent itinerary control is straightforward (the itinerary is restricted to the single destination host), this is not the case in multi-hop mobility, where slight variations on the set of visited hosts or even on the order that a specific set of nodes is visited may result in dramatic changes of the overall trip latency and migration traffic. In this article, we focus on multi-hop mobility, aiming at devising methods to optimise MA itineraries.

The remainder of the paper is organised as follows: Section 2 explains the importance of optimal agent itinerary planning in management applications. Section 3 discusses the background and the functionality of an algorithm that addresses this issue, while Section 4 focuses on the implementation details and Section 5 presents experimental results. Section 6 reviews research projects of relevance to our work and Section 7 concludes the paper and describes ongoing research.

2. IMPORTANCE OF ITINERARY PLANNING IN AGENT-BASED MONITORING

Despite the great potential of exploiting agent mobility in distributed applications, a naive use of MAs may lead to a highly inefficient design. In network monitoring applications for instance, using a single MA object launched from the manager platform that sequentially visits all the managed NEs, regardless of the underlying topology may actually lead to performance worse than the conventional SNMP-based approach (see Figure 1a). The performance further declines, when the monitoring MA collects monitoring data from multiple subnets, often interconnected by low-bandwidth links (see Figure 1b).

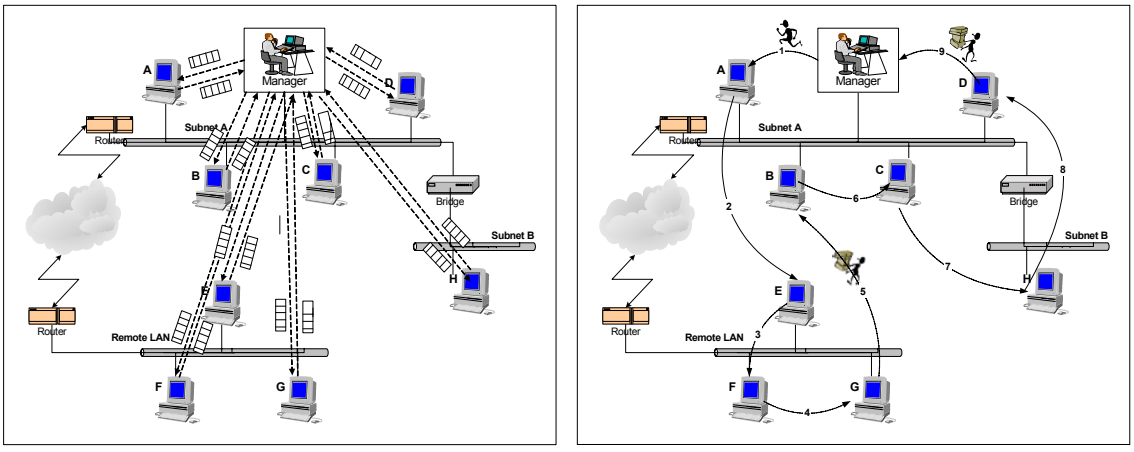

Figure 1. (a) Centralised monitoring, (b) 'Flat' MA-based monitoring

This inefficient approach, known as 'flat' MA-based monitoring [4], presents serious scalability problems: in large networks the *round-trip delay* of the MA greatly increases as the overall travel time depends on the number of hops realised by the MA and also the *network overhead* imposed by the MA transfers grows exponentially with the network size [4]; the slope of the overhead curve becomes steeper in the case of high selectivity[1] values.

A rational approach to overcome such scalability problems is to partition the managed network into several logical/physical domains. For instance, inFigure 2, an MA object polls the devices of the remote LAN, whereas a second MA is assigned to the subnet local to the manager host as well as to another subnet, which is part of the same LAN.

The partitioning criteria could be the number of nodes assigned to each MA (i.e. equal distribution of managed hosts among individual MAs), the physical distribution of polled devices (i.e. one MA object

[1] Selectivity σ ($0 \leq \sigma \leq 1$) is a metric defined in as the proportion of data *maintained* to that *retrieved* from each host. For low selectivity values (the major part of the obtained data being filtered at the source) the MAs' state size practically remains constant, otherwise the state rapidly grows [4].

assigned to each network segment), or a combination of the previous criteria. An analytical evaluation of the partitioning criteria effect on the performance of management tasks, is given in [4].

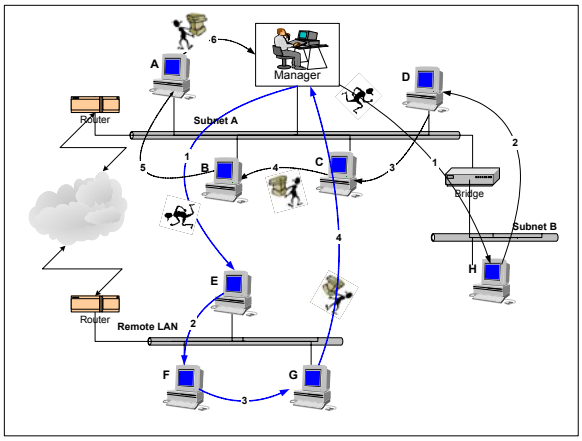

Figure 2. Partitioning the network into two distinct domains

Although partitioning the managed network into separate domains represents a step forward improving management scalability, it is clear that the scenario illustrated in Figure 2 represents an ideal case in terms of the WAN link utilisation. That is because the link is traversed only twice per polling interval (PI). For instance, network partitioning could be performed in such a way that the two MA objects would be required to visit NEs located in both the segment local to the manager station and the remote LAN. This is pictured in Figure3a, with the first MA following the itinerary BEF and the second ACGDH. In this case, the WAN link is traversed four times per PI. In a more pessimistic scenario, one of the MAs would not visit the remote LAN hosts in sequence, but traverse the link three times on each direction (itinerary: EBFCG); the second MA travel plan would be HDA (see Figure3b). Apparently, even when specific partitioning criteria are followed, the design of MA itineraries is almost random; itineraries scheduling process lacks a mechanism that would guarantee minimal use of links interconnecting individual management domains, hence *optimal itineraries planning* (OIP) is required.

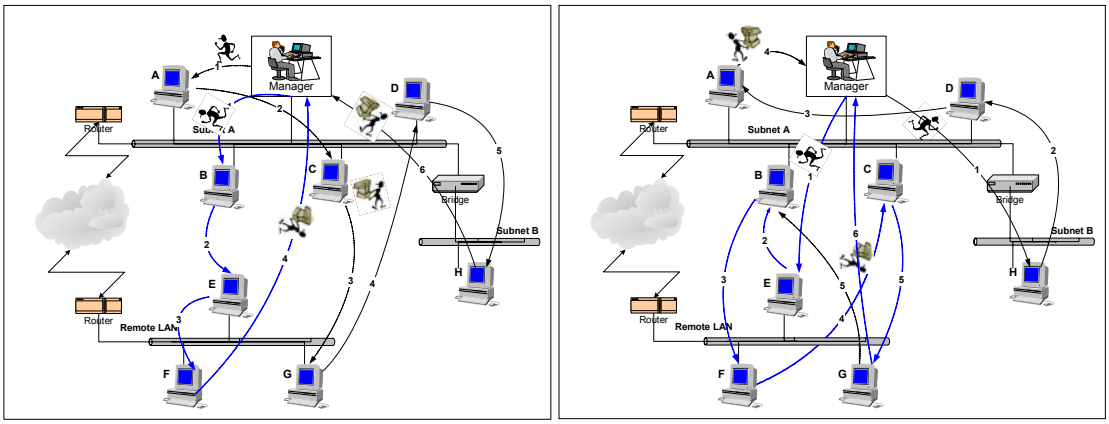

Figure 3. Non-optimised partitioning scenarios

3. THE OPTIMAL ITINERARY PLANNING ALGORITHM

Interestingly, the OIP problem exhibits many similarities with the Multi-point Line Topologies or Constrained Minimum Spanning Trees (CMST) problems. A CMST is a Minimum Spanning Tree[2] with the additional constraint on the size of the subtrees rooted on the center (there is an upper limit on the number of nodes included on each of the subtrees originated at the tree's root). CMST algorithms are used in graph theory, with the main application domain being network design problems. In such problems, the objective is the optimal selection of the links connecting terminals to concentrators[3] or directly to the network center,

[2] A Minimum Spanning Tree is defined as a tree (i.e. a connected graph without cycles) with the least total distance, cost, or some other metric of delay or reliability [7].

[3] Concentrators (or multiplexors) are nodes that consolidate low speed lines into higher speed lines, directed to the network center [7].

resulting in the minimum possible total cost. The output of CMST algorithms typically comprises topologies partitioned on several multi-point lines (or tree branches), where groups of terminals share a subtree to a specific node (center). For instance, Figure 4a depicts a set of nodes with a given network center and costs for connecting individual pair of nodes, while Figure 4b presents the optimal multi-line topology that minimises the overall cost, where network nodes have been partitioned into two clusters or subtrees, each directed to the network center. In this particular scenario, the overall cost will comprise the sum of costs for connecting each link included in the problem solution:

$$c = c_{5,3} + c_{3,1} + c_{1,0} + c_{4,2} + c_{2,0} = 35.$$

Since CMST problems are NP-hard, several heuristics have been developed to efficiently deal with them; *Esau-Williams* (E-W) and *Sharma*'s algorithms are the two most well known [7].

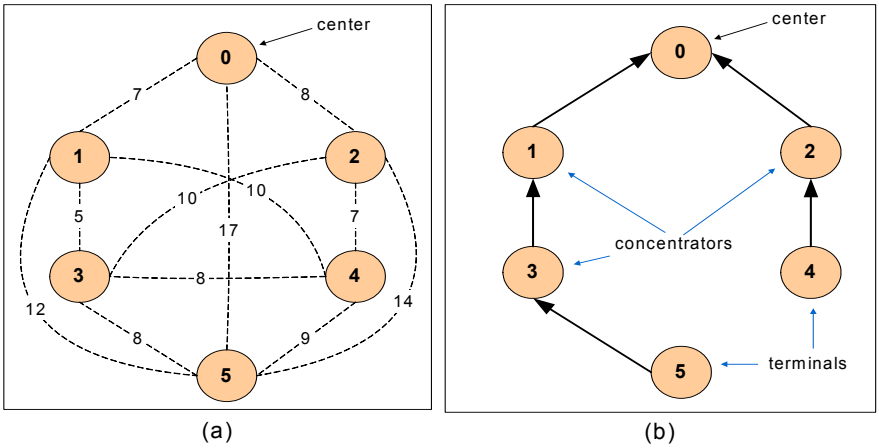

Figure 4. CMST problem: (a) The unconnected graph; (b) The optimal multi-point line topology

The input of E-W algorithm comprises the number N of terminals to be connected, the index of the central network site, constants $w_{i,j}$ and $c_{i,j}$ $(1 \le i, j \le N)$ that denote the traffic (weight) requirement and the cost for connecting terminals i and j respectively, and finally the aggregate cost w_{max} that sets a maximum limit for the traffic routed through each individual multi-point line (subtree)[4]. The algorithm's output is a two-dimensional array including the endpoints (node indices) of the links included in the solution [7].

CMST algorithms are highly relevant to the OIP problem: partitioning a set of network terminals into multi-point lines is not semantically different to partitioning a set of managed devices into several domains. Besides, in both cases the objective remains to minimise the overall cost. The fact that CMST algorithms not only propose optimal clustering of the network into separate domains, but also decide on how the nodes included in each domain are interconnected (forming multi-point lines or subtrees) solves the problem of optimising the order that each MA visits the devices into its assigned domain: the MA object will simply have to traverse its respective subtree.

In principle, our OIP algorithm derives from the E-W algorithm. That is, we have adapted the parameters described above (defined using network design problems terminology) to reflect the specific characteristics of the OIP problems. The prototype of the method implementing the OIP algorithm (in Java) and the definition of the parameters involved, follow:

public int[][] OIP (int N, int center, int c[][], int w[], int w_{max})

- *N*: the overall number of managed devices (to be visited by MAs);
- *center*: the index of the manager host;

[4] The Esau-Williams algorithm evaluates a tradeoff function $t_{i,j}$, associated with each link (i,j), defined as follows: $t_{i,j} = c_{i,j} - c_{c_i}$, where c_{c_i} is the cost of connecting the component containing node i to the center [7]. Namely, it connects a node directly to the center only when that represents a solution 'cheaper' than connecting it to the nearest set of nodes already interconnected.

- c: the cost matrix (dimensions: $N \times N$) determining the cost of agent migration between each pair of hosts;
- w: the amount of data collected from each visited device (dimension: N);
- w_{max}: the maximum number of nodes that may be visited by a single MA.

The function's output is a two-dimensional array including the endpoints of individual agent hops (migrations) included in the solution.

In order to execute the OIP function, an important task is to assign specific values to the parameters listed above, reflecting the topology characteristics of the examined managed network.

In particular, the managed network V is defined as a union of its s individual subnets: $V = \{S_o, S_1, .., S_s\}$. The manager station, denoted as terminal *0* and located in subnet S_0 is appointed as the center of the managed network (graph). Without loss of generality, the weight variables are set to: $w_i = 1, \forall i$, in order to eliminate their effect on the proposed solution. Namely, we make the assumption that the amount of data collected from each host is constant. In addition, the cost variables are set to:

$$c_{i,j} = \begin{cases} 0, \forall\, i, j \in S_k \\ c(k,l), \forall\, i \in S_k, j \in S_l, k \neq l \end{cases} \qquad \textbf{(3-1)}$$

Therefore, for hosts *i* and *j* located in separate subnets the cost $c_{i,j}$ of an MA migration from *i* to *j* is not constant but depends on the actual 'distance' between their respective subnets, i.e. the number of intermediate subnets that need to be traversed, the bandwidth of their interconnecting links, etc. Moreover, the cost for traversing each link is proportional to the inverse of its bandwidth. That is:

$$c(k,l) = k * \left(\frac{1}{B_{S_k,S_1}} + \frac{1}{B_{S_1,S_2}} + \ldots + \frac{1}{B_{S_n,S_l}}\right) \qquad \textbf{(3-2)}$$

where S_1, S_2, … S_N are the subnets physically located 'between' subnets S_k and S_l (traffic between S_k and S_l is routed through them), B_{S_1,S_2} denotes the bandwidth of the link connecting S_1 and S_2 and k is a normalisation factor.

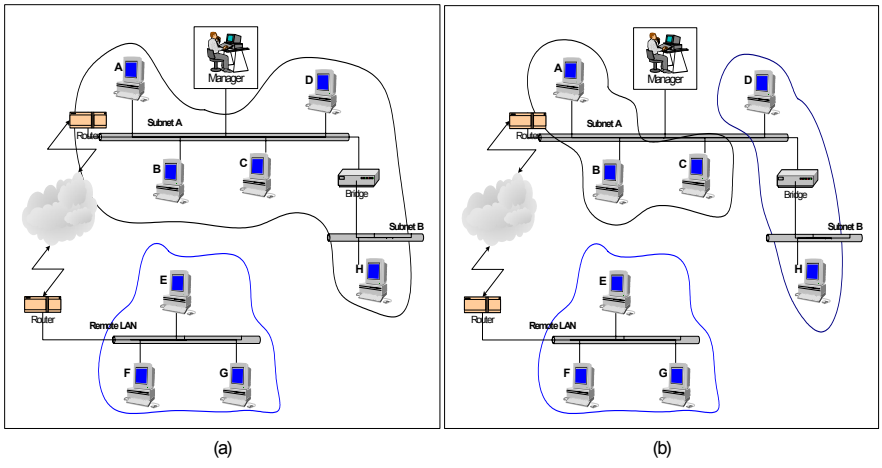

Figure 5. Applying an OIP heuristic to propose (a) two, or (b) three near-optimal itineraries.

Using as case study the simple network topology illustrated inFigure 2, the employment of the OIP algorithm prioritises the inclusion of the hosts located on the remote LAN into a separate management domain, for both the cases that two or three optimal MA itineraries were requested (see Figure 5). In the extreme case that only one itinerary is requested, the application of the OIP algorithm ensures that the remote LAN hosts are visited successively, namely the WAN link is being traversed only twice. Therefore, the algorithm described above not only provides optimal clustering of the managed network in management domains, but also optimal solutions regarding the order in which individual MA objects should visit their assigned NEs.

4. ALGORITHM IMPLEMENTATION DETAILS AND PERFORMANCE EVALUATION RESULTS

The configuration of the OIP algorithm parameters is a task undertaken by the human administrator, through the GUI pictured in Figure 6.

First, the criterion according to which the managed network will be partitioned into separate managed domains is chosen. Should the administrator chooses to grant the management framework the authority to automate the partitioning phase and construct optimal itineraries for the MAs involved in the monitoring process (execution of the OIP algorithm), two additional options are given: (a) Let the OIP algorithm decide on the optimal number of itineraries as well as the number of devices included in each managed domain; (b) Define an upper limit on the number of devices that may be included in each domain (that will certainly affect the final number of managed domains). If the latter option is chosen, the administrator defines the maximum number n of devices; the OIP function will then be invoked, setting the parameter $w_{max} = n$. If the former option is chosen, the parameter will be set equal to the total number of devices: $w_{max} = N$.

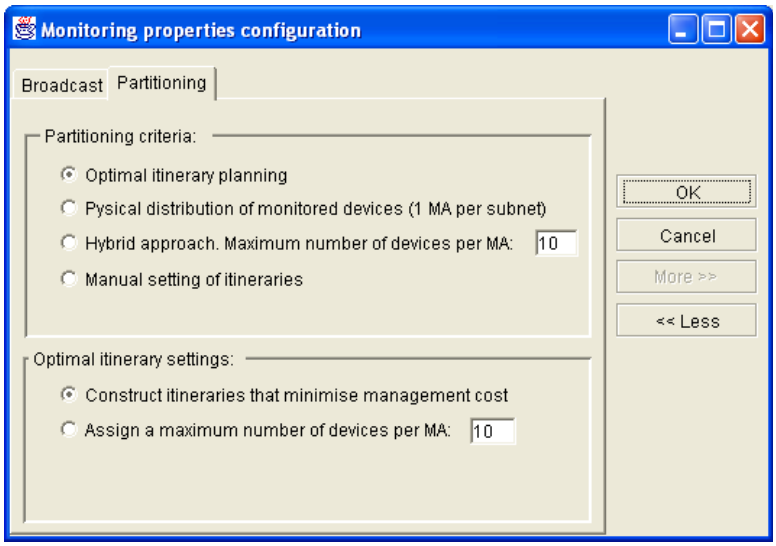

Figure 6. Configuration of OIP algorithm parameters

The results of several algorithm tests (for varying network topologies) revealed that OIP algorithm favors the 'connection' of NEs local to the manager subnet directly to the manager host (i.e. single-hop agent itineraries). Such solutions are clearly inefficient though as they result in the transmission of large numbers of MAs affecting the overall scalability. That is because the algorithm lacks a mechanism that would make the 'connection' of these NEs directly to the manager host less appealing than choosing to 'inter-connecting' them and forming a subtree. In order to exclude such results from the set of valid solutions, we have chosen to slightly modify the cost matrix of equation (3-1), introducing a small cost penalty for 'connections' between the manager host and devices residing on its local subnet. An alternative way around this problem would be to introduce an additional constraint w_{min}, which would prevent the construction of single-hop itineraries (by setting $w_{min} \geq 2$). However, this idea has been disqualified as in several topology scenarios, it failed to provide valid problem solutions.

Thus, the output of OIP algorithm for the test network of Figure 2 is depicted in Figure 7a, which demonstrates the optimal clustering of the managed network in two separate 'subtrees', each assigned to an itinerant agent.

Yet, there still remains an open issue to be addressed. There are three ways for an MA to traverse its domain (i.e. its subtree):

- Pre-order traversal (visit the root, then the left subtree, then the right subtree);
- In-order traversal (visit left subtree, then the root, then the right subtree);
- Post-order traversal (visit left subtree, then the right subtree, then the root).

Post-order traversal is the most efficient though, as it leaves the devices local to the manager site to be visited at the end of the MA's itinerary. That ensures that when traversing the link interconnecting two subnets, the MA has not yet collected any data, hence, it will have the minimum possible impact on network

resources. For instance, in Figure 2, the MA assigned to the domain including subnets A and B, first visits host H residing in subnet B and then returns to visit hosts D, C, B, A, which are local to the manager site.

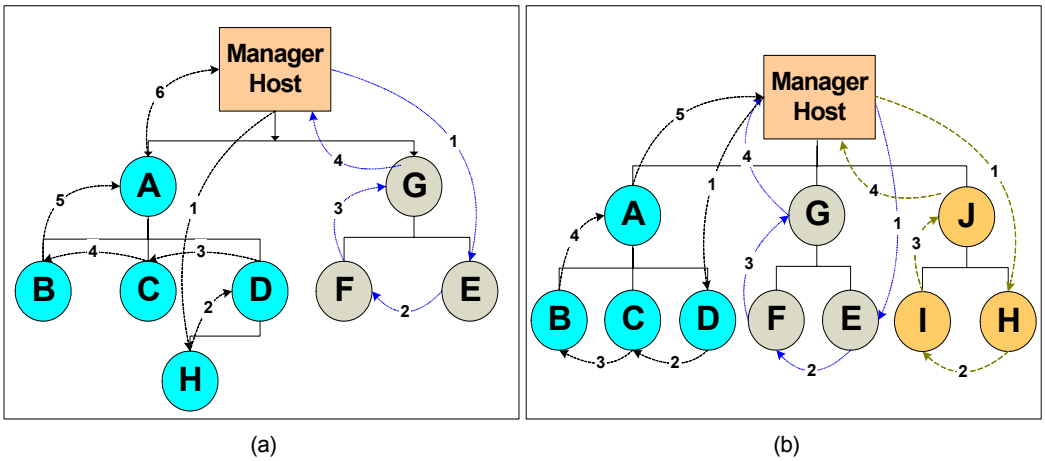

Figure 7. Output of the OIP algorithm: (a) For the test network of Figure 2, (b) After discovering two additional devices on Subnet B.

To integrate OIP process into our hierarchical management framework prototype introduced in [5], we have implemented a software component termed the *Itinerary Scheduler Module* (ISM). ISM executes the OIP algorithm and informs the manager application on the number of MAs that need to be instantiated and their respective itineraries. It also ensures that each agent itinerary represents a post-order traversal of its assigned 'subtree'. The manager application asks the ISM (through a callback function) to recalculate the optimal agent itineraries whenever a new managed device is 'discovered'. This is depicted in Figure 7b, which shows the output of the OIP algorithm when two additional devices are discovered on Subnet B (see Figure 8a).

In case that the number of managed devices on remote subnets becomes too large and given that specific criteria are met, the manager dynamically adapts to the new topology facts and deploys an MA (termed Mobile Distributed Manager [5]) to the remote subnet to act as a local manager thereby localising management traffic and reducing the associated cost (see Figure 8b).

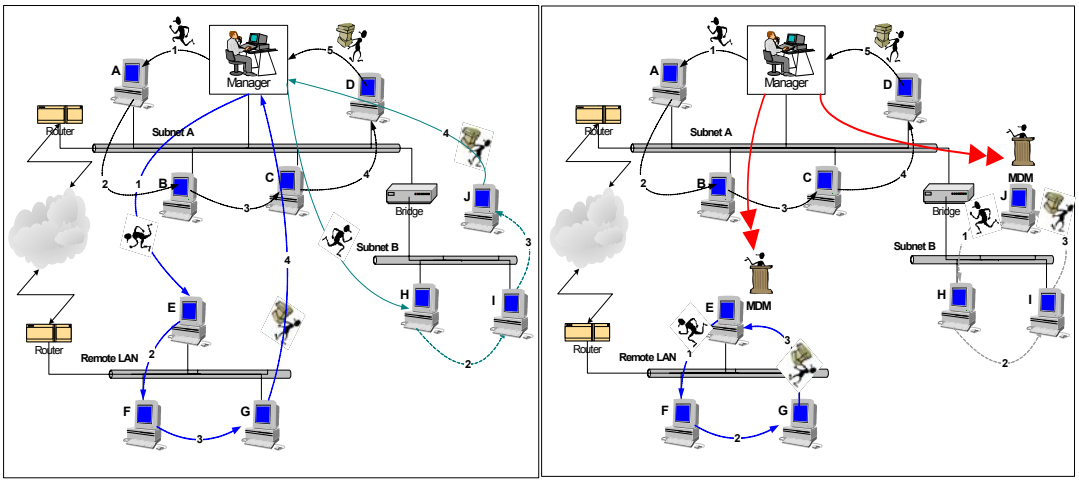

Figure 8. Discovery of new managed devices: (a) Reconfiguring optimal itinerary planning, (b) Applying an adaptive hierarchical MA-based management model

5. EXPERIMENTAL RESULTS

OIP algorithm performance tests have taken place on the network illustrated on Figure 8 comprising ten different WinNT machines with Pentium III (450 MHz) processor and 128MB of memory. The cost matrix has been calculated through equations (3-1) and (3-2), where $B_{S_A S_B}$ =10Mbps, $B_{S_A S_R}$ =1Mbps and k = 10^7. The MA used in our experiments has code size of 1.95 Kbytes, initial state size of 447 bytes and accumulates 50 bytes of data from each visited host. The algorithm has been tested on managed networks of 10 and 20

devices (in the latter case, two separate agent servers were simultaneously running to 'simulate' two managed devices on the same host). These managed networks have been segmented into 1, 2, 3 and 4 separate partitions. In the following table we present the average improvement measured when applying the OIP algorithm against another technique, where managed devices are equally distributed among a fixed number of domains; in the latter case, after determining the hosts comprising each management domain, the itineraries of the agents assigned to each domain are randomly constructed. It is noted that each table entry represents the average improvement measured over 100 runs: clearly, the OIP algorithm provides the same solution for specific sets of nodes and a given pair of (#devices, #partitions); the 100 runs refer to different, randomly generated itineraries. It can be noticed that the prevalence of the OIP algorithm is more evident when applying it to larger sets of nodes and smaller number of partitions (the optimal construction of agent itineraries then becomes a decisive factor).

		# partitions (managed domains)			
		1	**2**	**3**	**4**
# managed devices	**10**	37%	34%	21%	4%
	20	66%	52%	28%	6%

Table 1: Decrease of management cost when applying OIP algorithm over randomly generated itineraries

6. RELATED WORK

Iqbal et al. [6] developed a performance model that, given a specific communication pattern, allows agents to decide whether they should migrate to a site and communicate locally or the communication should be performed remotely. The decision is taken according to an Optimal Design Graph; in most cases, it has been indicated that the optimal performance of an agent is achieved by a critical sequence of mixed remote procedure calls and agent migrations.

A work conceptually relevant to the research presented herein, has been presented in [11], where Qi and Wang propose the employment of MA paradigm for data fusion in wireless sensor networks. To optimize agents itinerary, they derived a Local Closest First (LCF) algorithm according to which each MA searches for the next destination with the shortest distance to its current location. However, their cost function formulation does not take into account potential clustering of visited hosts in multiple physical subnets, which would affect the calculation of the distance matrix. Also, LCF-like algorithms have been characterized as 'nearsighted', in the sense that their output highly depends on the MAs original location, while the nodes left to be visited last are associated with high migration cost [7]; the reason for this is that they search for the next destination among the nodes adjacent to the MA's current location, instead of looking at the 'global' network distance matrix.

Most importantly though, both the works presented in [6] and [11], deal with the problem of constructing near-optimal MA itineraries for given sets of network nodes, where a *single* MA visits the whole set. Yet, they do not address the fundamental problem of partitioning network nodes in optimal clusters. On the other hand, the algorithm presented in the preceding sections deals with the optimal clustering problem and subsequently uses the algorithm's output to easily construct optimal agent itineraries (through the post-order tree traversal described above).

7. CONCLUSIONS & ONGOING RESEARCH

Despite the popularity of MA-based management applications, methodologies for designing efficient MA itineraries have received little attention. In particular, the number of hops realised by a multi-hop agent is not the only metric to evaluate the communication overhead of MA-based operations. The order in which MAs visit their assigned NEs is also a crucial factor, as slight changes on the agents itineraries may result in dramatically variant management costs.

This article introduces an algorithm that borrows ideas and concepts from the area of network design to address the issue of MA optimal itinerary planning. The algorithm has been designed having network monitoring applications in mind, although it would comfortably fit in other application areas, such as network discovery, service management, etc. The OIP algorithm not only suggests the appropriate number of

MAs that should be employed to minimise the overall management cost but also constructs optimal itineraries for each of them.

Ongoing research includes exhaustive testing of the presented OIP algorithm in large-scale, simulated, multi-subnet network environments using a network simulator tool. We also plan to test the applicability of the OIP algorithm in several application fields, other than network monitoring.

REFERENCES

[1] C. Bohoris, A. Liotta, G. Pavlou, "Mobile Agent Based Performance Management for the Virtual Home Environment", Journal of Network and System Management, 11(2), pp. 133-149, June 2003.

[2] T. Du, E. Li, A.P. Chang, "Mobile Agents in Distributed Network Management", Communications of the ACM, 46(7), July 2003.

[3] A. Fuggeta, G.P. Picco, G. Vigna, "Understanding Code Mobility", IEEE Transactions on Software Engineering 24(5), pp. 346–361, 1998.

[4] D. Gavalas, "Mobile Software Agents for Network Monitoring and Performance Management", PhD Thesis, University of Essex, UK, July 2001.

[5] D. Gavalas, D. Greenwood, M. Ghanbari, M. O'Mahony, "Hierarchical Network Management: A Scalable and Dynamic Mobile Agent-Based Approach", Computer Networks, 38(6), pp.693-711, April 2002.

[6] A. Iqbal, J. Baumann, M. Straßer, "Efficient Algorithms to Find Optimal Agent Migration Strategies", Universität Stuttgart, Fakultät Informatik, Bericht Nr. 1998/05, April 1998.

[7] A.Kershenbaum, "Telecommunications Network Design Algorithms", McGraw-Hill, 1993.

[8] A. Liotta, G. Pavlou, G. Knight, "Exploiting Agent Mobility For Large Scale Network Monitoring", IEEE Network, 16(3), pp. 7-15, May/June 2002.

[9] P. Marques, P. Simões, L. Silva, F. Boavida, J. Gabriel, "Providing Applications with Mobile Agent Technology", Proceedings of the 4th IEEE International Conference on Open Architectures and Network Programming (OpenArch'01), IEEE Computer Press, April 2001.

[10] A. Puliafito, O. Tomarchio, "Using Mobile Agents to implement flexible Network Management strategies", Computer Communications, 23(8), pp. 708-719, April 2000.

[11] H. Qi, F. Wang, "Optimal itinerary analysis for mobile agents in ad hoc wireless sensor networks", Proceedings of the 15th IEEE International Conference on Wireless Communications, pp.147-153, July 2001.

[12] J. Schönwälder, A. Pras and J.P. Martin-Flatin, "On the Future of Internet Management Technologies". IEEE Communications Magazine, Vol. 41, No. 10, pp. 90-97, October 2003

[13] W. Stallings, "SNMP, SNMPv2, SNMPv3 and RMON 1 and 2", 3rd ed., Addison Wesley, 1999.