Benchmarking Mobile Agent Itinerary Planning Algorithms for Data Aggregation on WSNs

Ioannis E. Venetis, Grammati Pantziou
Dept. of Informatics
Technological Educational Institution of Athens
Athens, Greece
{ivenetis, pantziou}@teiath.gr

Damianos Gavalas
Dept. of Cultural Technology and Communication
University of the Aegean
Mytilene, Greece
dgavalas@aegean.gr

Charalampos Konstantopoulos
Dept. of Informatics
University of Piraeus
Piraeus, Greece
konstant@unipi.gr

Abstract— **Mobile agents (MAs) have been proposed as a distributed middleware approach suitable for autonomic data aggregation operations in Wireless Sensor Networks (WSN). Determining optimal itineraries for MAs traveling through WSN nodes is a non-trivial problem. Thus, several heuristics have been proposed to perform efficient itinerary planning for MAs. However, the evaluation of these heuristics is typically performed on the ground of different parameter spaces and assumptions about the underlying network and the capabilities of nodes. Herein, we implement, simulate and compare the most prominent itinerary planning algorithms upon a common parameter space, making realistic network-level assumptions.**

***Keywords*— WSN, Mobile Agent, Data Aggregation, Single-Itinerary Planning, Multiple-Itinerary Planning.**

I. Introduction

Wireless Sensor Networks (WSNs) consist of a set of sensor nodes (SNs), i.e., devices able to measure aspects of the surrounding environment. Such networks provide the possibility of collaborating processing [1], a feature that allows data to be processed at and combined from multiple sources. It is quite common for data originated from neighboring sensors to have spatiotemporal correlation. Therefore, it is possible to perform some kind of aggregation on each sensor, as data passes from one sensor to the next one, until they reach a destination node, known as the Processing Element (PE) or sink. This reduces the total amount of data that has to be transferred, which leads to faster collection of data as well as reduced use of energy and network resources that would be otherwise spent for transmitting larger amounts of data.

Mobile Agents (MAs) have been proposed to implement data aggregating is WSNs. A MA is an autonomous program moving from node to node and acting on behalf of the users toward the completion of an assigned task. The software logic is carried with the MA to each SN and it determines the processing to be performed on each SN. The resulting data after the local data aggregation is then embedded within the MA's state and carried to the next SN, where the MA resumes execution and performs aggregation upon the data retrieved thereon.

The use of MAs for aggregation requires the definition of SNs visiting order, i.e., an itinerary has to be scheduled. The chosen itinerary largely affects the overall energy consumption and aggregation cost. As a result, a number of solutions have been proposed in order to minimize these costs. The literature includes algorithms that either dynamically determine the route of the MA by deciding on the fly the next SN to be visited or approximate statically the optimal MA route through heuristics. A dynamic approach is more suited for target tracking/ classification applications, wherein the trajectory of a moving object is initially unknown. Static itineraries are more suitable for data monitoring applications, where measurements of physical quantities (e.g., humidity, temperature, etc.) are periodically gathered and sent to the PE.

Both approaches have advantages and disadvantages. A MA with a dynamic itinerary can respond to faults that might occur during its migration simply by changing its itinerary on the fly. However, the dynamic approach requires more time, since the decision about the next node is taken at each SN. It also consumes more energy on each SN and typically leads to larger MA sizes (the more intelligence integrated within the MA, the larger its size). Furthermore, an efficient routing protocol should be implemented in SNs so that MAs can determine their next hop as well as their return path to the sink. This is not needed when using static itineraries as each MA uses a predetermined itinerary that is calculated a priori at the PE.

Static itinerary planning algorithms are classified in Single-Itinerary Planning (SIP) and Multi-Itinerary Planning (MIP) algorithms [2], i.e., either a single MA is sent to collect the data or more MAs are sent in parallel, each to a subset of the nodes of the WSN. SIP performance is satisfactory for small WSNs but it deteriorates as network size grows. This is because both the MA's roundtrip delay and the energy consumption increase fast with network size, as the traveling MA accumulates data from visited sensors. The growing MA's size results in increased consumption of the wireless bandwidth and the limited energy of SNs. On the other hand, MIP algorithms aggregate data from smaller subsets of the SNs, mitigating the effects of the SIP algorithms. However, they are more difficult to design and complex to execute, as they require to first group SNs into subsets in a way that will create itineraries of approximately equal cost.

Due to the considerable number of already proposed itinerary planning algorithms, it is rather difficult to compare the quality of their derived itineraries with respect to a set of metrics that determine the optimal behavior for a specific application. The situation is further complicated by the fact that performance evaluations for each algorithm are performed upon parameter sets, different than those used by others. As if this was not already enough, different assumptions are made about the capabilities of the SNs comprising the WSN.

The contributions of this paper are (a) a detailed performance evaluation and benchmarking of most prominent itinerary planning algorithmic approaches with respect to several performance indicators based on the common ground of realistic aspects, like increasing MA size, migration cost dependent on number of intermediate hops and availability of different levels of transmission power, (b) simulation of compared algorithms on a WSN-tailored simulator (Castalia [13]) which provides detailed implementations of MAC layer protocols as well as realistic energy models (taking into account energy spending for transmitting, receiving, sensing and remaining standby), hence allowing precise estimation of performance parameters (energy consumption, transmission delay, etc).

The paper is organized as follows: In Section II we briefly present the itinerary planning algorithms implemented along with their basic functional characteristics. In Section III we discuss the assumptions made and the common realistic parameters used in our benchmarks to fairly compare the examined algorithms. In Section IV we present and discuss the results of the performance evaluation. Finally, Section V concludes the paper.

II. Mobile Agent Itinerary Planning Algorithms

For the purposes of this paper, a total of eleven (11) algorithms have been implemented and evaluated on a common set of parameters in the Castalia simulator [14]. It is noted that all algorithms determine the itineraries of MAs statically. Out of these algorithms, 6 are SIP and 5 are MIP. In the rest of this section we briefly describe these algorithms.

Qi and Wang [2] proposed two heuristics to optimize the itineraries of MAs performing data aggregation tasks. In the Local Closest First (LCF) algorithm, each MA starts its route from the PE and searches for the next destination with the shortest distance to its current location. In the Global Closest First (GCF) algorithm, MAs also start their itinerary from the PE and select the SN that has not yet been added to the itinerary and is closest to the center of the surveillance region as the next-hop destination.

The output of LCF highly depends on the MA's original location, while the SNs left to be visited last are typically associated with high migration cost. The reason for this is the next destination is greedily sought among the SNs adjacent to the MA's current location, instead of looking at the "global" network distance matrix. On the other hand, GCF produces messier routes than LCF and repetitive MA oscillations around the region center, resulting in long route paths and unacceptably poor performance [2], [3].

The above algorithms assume a full aggregation model, i.e., feasibility to aggregate data into a single constant size packet. However, in most practical scenarios MAs grow heavy while moving along their itinerary; hence, it is important select low-cost links (i.e. those requiring low transmission power) for the last part of the itinerary, when MAs have become larger. Failing to recognize this fact might result in much higher energy spending, especially on the last MA hops. In order to verify this conclusion, our implementation of the LCF and the GCF algorithms assumes partial data aggregation, making their comparison with the rest of the algorithms fairer.

A slightly different approach is taken in the Mobile Agent-based Direct-Diffusion (MADD) algorithm [4]. Although quite similar to LCF, MADD starts by selecting as the first SN in the itinerary the furthest one from the PE. From that point on, its selection strategy for the next SN in the itinerary is the same as in LCF. In contrast to LCF and GCF, MADD assumes a partial aggregation model.

More recent variants of the LCF are the Itinerary Energy Minimum for First-source-selection (IEMF) algorithm and the Itinerary Energy Minimum Algorithm (IEMA), which is an iterative version of IEMF [5]. LCF is a greedy algorithm, at every SN selecting the closest, unused SN as the next stop in the itinerary. The quality of itineraries derived by LCF very much depends on the selection of the first visited node. To address this issue, IEMF considers all SNs as candidate first nodes and performs LCF thereafter, finally selecting the minimum cost itinerary. IEMA extends this idea by recursively selecting the first k nodes that minimize the cost of the itinerary. As shown in [5], when k becomes 20% of the total nodes, IEMA achieves a fairly good estimation of the minimum cost. Keeping k as low as possible reduces the total time required by the algorithm to plan the itinerary. For the implementation of IEMA herein, we have used the above percentage of the total number of nodes.

Although not specifically tailored towards WSN applications, heuristics solving the Travelling Salesman Problem (TSP) can be used to formulate a solution for the itinerary planning problem. If every SN is thought of as a city, the distances between SNs can be used in well-known heuristics that solve the TSP. In the context of this paper, we used the Lin-Kernighan heuristic as described in [6] to create a single itinerary for a MA. It is noted that the TSP-based algorithm has not been proposed in the literature and has been implemented mainly to serve as a comparison reference point.

Apart from the SIP algorithms described in the previous paragraphs, a number of MIP algorithms have also been implemented. The basic difference is that the total set of SNs is firstly grouped in disjoint clusters according to some heuristic; then, each cluster is assigned to an MA and a SIP algorithm is employed to determine the itinerary of each MA through its assigned nodes.

The Visiting Central Location (VCL) algorithm [7] uses the notion of an impact factor to determine a set of SNs that will act as central points in each cluster. The impact factor is defined as a value that measures how much each SN is affected by the presence of other SNs. In the context of [7], a function similar to the calculation of gravitational forces among bodies is used. The SN with the highest impact factor is selected as the center of the next cluster. All SNs located within a specific distance are then attached to the cluster of that SN. The procedure repeats with the remaining SNs, until all SNs belong to a cluster. A parameter σ is used to determine how strong an SN impacts other SNs. As in [7], we set $\sigma = 8$ in this paper.

Another MIP approach is the Balanced Minimum Spanning Tree (BST) algorithm [8]. Prim's algorithm is executed to determine a spanning tree, rooted at the PE. Nodes of a single branch in the tree are considered to belong to the same cluster. A SIP algorithm is then executed to decide on the order in which cluster nodes will be visited. In our implementation, we used IEMA. Furthermore, a parameter α is used in BST to balance the weights calculated during Prim's algorithm. The idea is to create more balanced branches, i.e., each branch should contain as many SNs as all other branches. In [8] it is shown that $\alpha = 0.6$ is a good choice; therefore this is the value we use as well.

The Directional Source Grouping (DSG) algorithm [9] uses a circle around the PE. The radius of the circle is set to the maximum transmission range of a single SN. Every SN that lies in the circle is used as the first node of an itinerary. Clusters are built by selecting SNs that lie within a circular sector and a SIP algorithm is used to decide the order of visiting each SN in the cluster. In our implementation, the size of each sector is determined by an angle θ, which in turn is calculated from the number of SNs that lie in the circle defined above ($\theta = 2 \cdot \pi$ / (Number of SNs in circle)). Furthermore, while defining the sectors, a methodology similar to the CL algorithm is followed. For this part of the DSG algorithm we use the same parameters as for VCL.

The Near-Optimal Itinerary Design (NOID) algorithm [10] adapts a method originally designed for network design problems, namely the Esau-Williams heuristic [13] for the Constrained Minimum Spanning Tree (CMST) problem, in the specific requirements of WSNs. NOID recognizes that MAs aggregate data while visiting SNs without returning back to the PE to deliver their collected data. Therefore, NOID restricts the number of migrations performed by individual MAs, thereby promoting the parallel employment of multiple cooperating MAs, each visiting a subset of SNs. In particular, NOID iteratively adds SNs to separate subtrees, which progressively converge towards the PE. Each subtree is then assigned to a separate MA, which performs a post-order traversal of its subtree so as to visit first the SNs residing furthest from the PE.

The Tree-Based Itinerary Design (TBID) algorithm [12] improves upon NOID by following a more direct approach to the problem of determining low-cost MA itineraries. Specifically, based on an accurate formula for the total energy consumed during MA migration, a greedy approach is followed for distributing SNs in multiple MA itineraries. Essentially, the algorithm determines a spanning forest of trees in the network, calculates efficient tree traversal orders (itineraries), and eventually, assigns these itineraries to individual MAs. Not only does TBID assume a general aggregation model, where data after aggregation does not necessarily have constant size, but it actually exploits this fact during decision making. As a result, TBID always selects low-cost links for the last hops of itineraries, where the energy consumption due to increased data traffic is expected to be high. Moreover, the partitioning of SNs in multiple itineraries reduces the maximum load each MA carries along its itinerary. This again leads to lower energy expenditure in the last itinerary legs. TBID is built around the idea of co-centric zones with the PE as the center. The first zone includes all SNs that are within a certain distance of the PE and a MA is assigned to each itinerary rooted at those SNs. In this paper the radius of the first zone is set to the maximum transmission range of a single SN.

III. Assumptions and Simulation Parameters Used for Benchmarking Itinerary Planning Algorithms

From the discussion in the previous section, it becomes evident that the path design phase of most itinerary planning algorithms is based on a number of assumptions that are not always valid in real WSN deployments. The most noteworthy example is the fact that the decision on selecting next MA hops is typically based on geographical distance criterion. This implies that the nodes can communicate directly, with the MA migration cost increasing linearly with distance. However, in real environments two nodes can only communicate if they lie within a certain distance. Namely, most algorithms do not take into account the transmission range of nodes when taking route planning decisions. In real environments, this would lead to infeasible migrations, when considering long hops. To make migrations feasible, a number of intermediate nodes would have to be visited on the way from a source SN to another. It is noted, though, that the number of intermediate nodes is not necessarily equivalent to the distance among the end nodes, especially when considering sparse network topologies. The above is one of the reasons why most algorithms assume dense network topologies.

In this article, we take into account all restrictions encountered in real settings, yet, without distorting the basic logic of examined algorithms. In particular, we allow each algorithm to decide 'as usual' about the next node to be included in a path. If, however, the next node is out of range, the path of the MA includes intermediate nodes, ensuring that nodes visited in sequence are within transmission range. Namely, we clearly distinguish among source and intermediate nodes, with the former being visited to retrieve sensory data and the latter serving as in-between stops while migrating from a source node to another. The intermediate path is the shortest path found among the two end nodes (constructed using Dijkstra's algorithm); the cost of each edge along the path equals the transmission power level required for direct communication (edges only connect nodes which lie within transmission range). This approach allows all algorithms to act as if any two nodes in the network can communicate directly. Furthermore, all implemented algorithms have been enhanced to disregard clusters of nodes where every node in the cluster has no path to the PE. Hence, we relax the requirement for the algorithms to be applied on dense networks. Since a MA starts from the PE, only the nodes that can be accessed from a path that includes the PE are taken into account.

Another element commonly found in the performance evaluation of investigated itinerary planning algorithms is the fact that the radio module of sensor nodes is oversimplified, disregarding several realistic restrictions. Firstly, transmission ranges are typically defined to be larger than real radio devices for WSN nodes can achieve. Secondly, most real radio modules feature a number of different transmission levels, with each level consuming different amounts of energy and transmitting to different ranges. In the context of this paper, we used the Castalia simulator which simulates real radio modules at a detailed level. Furthermore, whenever a transmission has to be performed between two nodes, the transmission level of the sending SN is set to the lowest possible level that will allow direct communication between the nodes. This allows more accurate estimation of the energy consumed during data aggregation operations, compared to using a single (highest) power level.

Finally, a last reason that makes it difficult to compare existing itinerary planning algorithms is that they are typically tested using different parameters during simulations. These parameters concern the number of network nodes, the terrain size, the transmission range of nodes, the energy consumption for receiving and transmitting data, the energy consumptions to execute an MA, the size of data accumulated by MAs from each node, the execution time of MA at each node, etc. In this article, a common set of testing parameters is used, most corresponding to values taken from commercial radio modules. This makes the presented results more consistent and reliable.

IV. Results

We have implemented 11 MA itinerary planning algorithms and evaluated their performance under the realistic assumptions detailed in the previous section and using a common parameter space. The implementations have been tested on the Castalia simulation platform. A number of sensor nodes have been randomly deployed on a 500m × 500m area. A total of 10 deployments have been tested per network size (each algorithm has been simulated on the same 10 random deployments to ensure fairness). In the results discussed later on, we present the averages over all deployments. The PE is positioned at the center of the simulated terrain.

Each sensor node is assumed to be equipped with a CC2420 chip, a true single-chip 2.4 GHz IEEE 802.15.4 compliant RF transceiver designed for low power and low voltage wireless applications. The Castalia simulator is in charge of simulating the radio module. TABLE I. presents the eight available transmission levels of the CC2420, along with the required power to transmit data at a specific level and the distance that can be covered during transmission. While transferring an MA from one node to another the lowest possible transmission level is used to cover the required distance. Further parameters related to energy consumption per node (e.g. while receiving data, switching states, etc) are handled by Castalia and are again common throughout all experiments.

TABLE I. The available transmission levels of the CC2420 radio module, along with the power consumed and the transmission range achieved at each level.

Transmission level (dBm)	0	-1	-3	-5	-7	-10	-15	-25
Transmission power (mW)	57.42	55.18	50.69	46.20	42.24	36.30	32.67	29.04
Transmission distance (m)	46.42	42.17	34.81	28.73	23.71	17.78	11.01	4.22

TABLE II. The number of nodes randomly deplayed on the simualted area and the number of sensors that report their position to the PE.

Nodes deployed	100	200	300	400	500
Nodes reported	20	160	298	399	499

We assume that each SN is aware of its exact location. At the beginning of each simulation test, each node tries to contact neighboring nodes and establish a path towards the PE (using its maximum transmission range) so that it sends to the PE its location information. The latter is needed to calculate the Euclidian distance, hence, the power level required for the communication among each pair of nodes. TABLE II. presents the number of nodes deployed in the area and how many manage to establish such a path. For topologies featuring 300 nodes or more, practically all nodes report their position to the PE. At 200 nodes already a substantial 80% of the nodes report their position. Although with 100 deployed nodes the percentage of nodes that participate in each algorithm is very low, we included this case to demonstrate that our implementations do not require a dense network of sensors but can adapt their operation to the reachable nodes.

TABLE III. presents some further parameters used for our simulations. The first two rows represent actual values for the

TABLE III. Simulation parameters used throughout all conducted experiments.

Simulation Parameter	***Value***
Power consumed while receiving data (per second staying in Rx mode)	62 mW
Network transfer rate	250 Kbps
Initial node energy	18720 J (2 × AA Batteries)
Energy consumed for MA execution (data aggregation)	5 nJ
Mobile agent instantiation delay	10 ms
Mobile agent processing delay	50 ms
Code size of MA	1024 bytes
Size of data collected at each node	200 bytes
Data aggregation coefficient	1.0

simulated CC2420 radio module. The values referring to MA-specific parameters have been set based on real test results published in the relevant literature. The data aggregation coefficient has been set to 1.0, thus, implying full data aggregation. The same set of parameter values have been used through the whole range of simulated algorithms, making the comparison among them straightforward.

Results of the simulated scenarios are presented in the figures that follow. Figure 1 illustrates the time required for an MA to visit all nodes which are part of the itinerary as calculated by each algorithm. This includes the time required to visit intermediate nodes. Isolated nodes (i.e. those with no path to the PE) are excluded. Not surprisingly, MIP algorithms obtain shorter service times, as multiple MAs are travelling along nodes in parallel, each following a shorter path. The service time in MIP algorithms is determined by the length of the longest itinerary. Hence, MIP algorithms performing worse are those that either fail to calculate the suitable number of MAs or derive rather unbalanced itineraries.

TBID yields the shortest service time, followed by NOID. It is also interesting to observe that TBID improves its service time as more nodes are added to the sensor network. This is because more sensors lie within the first zone which is directly accessible by the PE. As a result, more MAs are sent to the network, each assigned a shorter path to follow on average.

Another interesting observation is the fact that most SIP algorithms have marginal differences in their service time (apart from GCF, which clearly performs worse). It is also noted that the path calculated by TSP is the best among the SIP algorithms used, although it is not specifically designed to be used in this context.

Figure 2 shows the total energy consumption for all nodes that are part of an MA itinerary. As with the service time, energy consumption of intermediate nodes is taken into account. Although MIP algorithms generally prevail against SIP algorithms, their performance gain is not as clear as with service time. In fact, TSP outperforms several MIP algorithms. This result has been somewhat unexpected: MIP algorithms send out more MAs, each assigned fewer nodes. Hence, each MA should normally transfer less sensory data from node to node, especially compared to the last steps of SIP algorithms. This should have led to lower overall energy consumption.

The justification to the above paradox is provided by Figure 3, which demonstrates that MIP algorithms derive itineraries with substantially more intermediate nodes. Although no data is collected at these nodes, energy is consumed to receive the MA and retransmit it to the next node. As a result, the energy efficiency of MIP algorithms reduces. It is worth noting that TSP performs particularly well with respect to these two metrics. It has the lowest total energy consumption and the lowest number of intermediate nodes. This second observation further strengthens our argument about the importance of taking into account intermediate nodes in assessing the performance of examined algorithms.

Figure 4 illustrates the performance of compared algorithms as regards network lifetime, i.e., the time that the first node dies due to energy depletion (network lifetime is

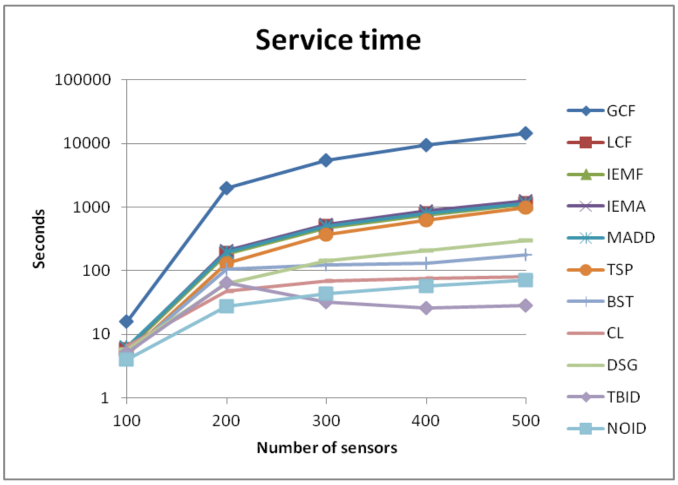

Figure 1. Time required (in seconds) for MAs to visit all source nodes included in their itinerary.

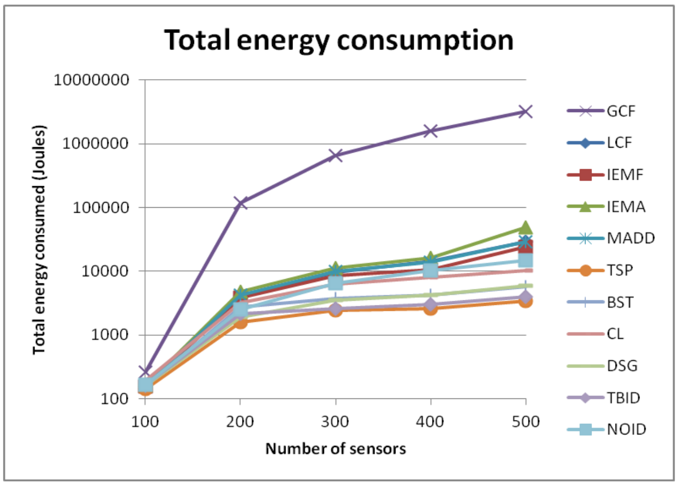

Figure 2. Total energy consumed (in Joules) by network nodes throughout a single data aggregation operation (i.e. due to transfer and execution of all MAs along their itineraries).

determined by the node that requires most energy to operate). Since the path calculated by an algorithm does not change during consecutive travels of the MA, the same node will always spent the maximum energy and it will be the first to deplete. SIP algorithms perform worse in this aspect. As more data is accumulated after visiting each source node, nodes towards the end of the path have to transmit large amounts of data to the next node. Due to the greedy logic shared among many of these algorithms, distances between nodes towards the end of the path tend to be larger, meaning that the maximal transmission level needs to be used. These two factors combined lead to increased maximum power consumption.

V. Conclusions and Future Work

Itinerary planning algorithms for mobile agents enrolled in data aggregation tasks in WSNs are pursuing to balance among several metrics that characterize their performance, like service time, energy consumption, network lifetime, etc. In order for a comparison among different itinerary planning

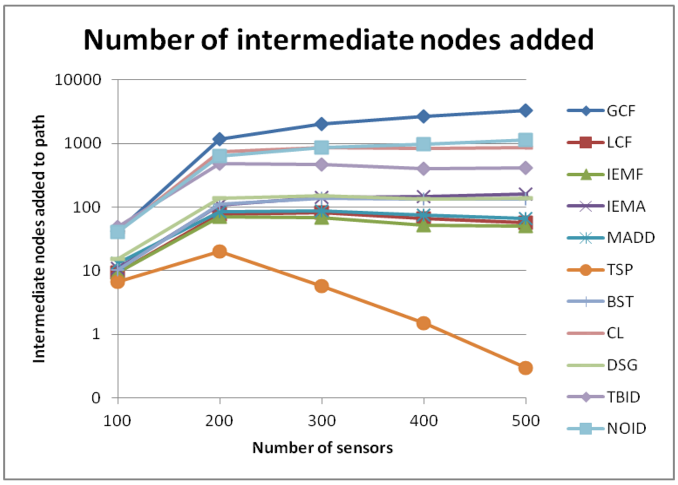

Figure 3. Average number of intermediate nodes included over all itineraries.

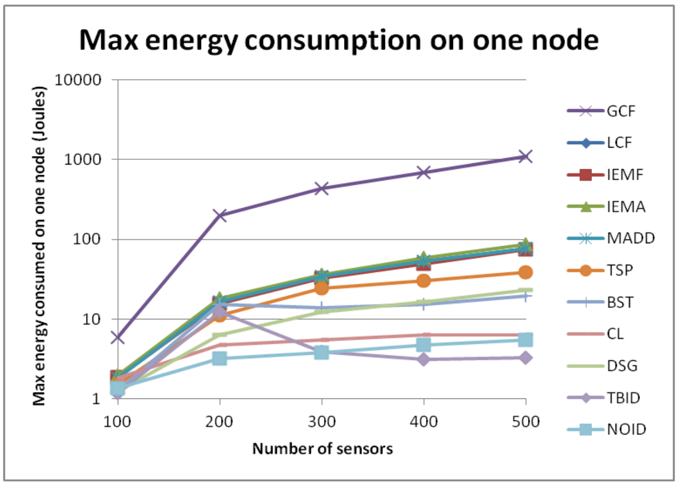

Figure 4. Energy consumed by the most heavily utilized node among all network nodes included in itineraries.

algorithms to be valid, two important aspects have to be taken into account. Firstly, the evaluation of the algorithms should be performed upon a common platform, utilizing the same set of parameters among those that affect their performance. Secondly, having derived the MA itineraries (namely, the ordered sets of source nodes assigned to individual MAs), the determination of the actual paths should take into account all restrictions inherent in real environments and, hence, include intermediate nodes in-between source nodes, if necessary.

In this work we implemented the most representative static itinerary planning algorithms and compared them following the aforementioned rules. Our simulation results reveal several interesting performance aspects. Although service time of MIP algorithms is significantly lower than SIP algorithms, the energy consumption of the former is not as good. The main reason is that their itinerary planning decisions lead to the inclusion of a substantially larger number of intermediate nodes along derived paths. This implies that the design of MIP algorithms explicitly incorporating transmission range of sensor nodes in itinerary planning decisions could possibly lead to including smaller numbers of intermediate nodes, hence, improved performance.

Nevertheless, the paths derived by MIP algorithms remain shorter than those produced by their SIP counterparts, which results in clear prevalence of MIP algorithms with respect to network life-time. Among all examined algorithms, TBID has been found to perform better, that is, to offer a better balance among all performance indicators.

VI. Acknowledgments

This research has been co-financed by the European Union (European Social Fund–ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) – Research Funding Program: Archimedes III. Investing in knowledge society through the European Social Fund.

References

[1] F. Zhao and L. Guibas, Wireless Sensor Networks. Morgan Kaufmann Publishers, 2004.

[2] H. Qi and F. Wang, “Optimal Itinerary Analysis for Mobile Agents in Ad Hoc Wireless Sensor Networks”. Proc of the 13th Int’l Conf. Wireless Comm. (Wireless ’01), Vol. 1, pp. 147-153, 2001.

[3] Q. Wu, N. Rao, J. Barhen, S. Iyengar, V. Vaishnavi, H. Qi, and K. Chakrabarty, “On Computing Mobile Agent Routes for Data Fusion in Distributed Sensor Networks”. IEEE Trans. on Knowledge and Data Eng., 16(6), pp. 740-753, 2004.

[4] M. Chen, T. Kwon, Y. Yuan, Y. Choi and V. C. M. Leung, “Mobile Agent-Based Directed Diffusion inWireless Sensor Networks”. EURASIP Journal on Advances in Signal Processing, 2006.

[5] M. Chen, L. T. Yang, T. Kwon, L. Zhou, and M. Jo, “Itinerary Planning for Energy-Efficient Agent Communications in Wireless Sensor Networks”. IEEE Trans. on Vehicular Technology, 60(7), pp. 3290-3299, 2011.

[6] K. Helsgaun, “An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic”. European Journal of Operational Research, 126(1), pp. 106-130, 2000.

[7] M. Chen, S. Gonzalez, Y. Zhang, and V. C.M. Leung, “Multi-Agent Itinerary Planning for Sensor Networks”. Proc of the IEEE 2009 Int’l Conf on Heterogeneous Networking for Quality, Reliability, Security and Robustness (QShine 2009), 2009.

[8] M. Chen, W. Cai, S. Gonzalez and V. C.M. Leung, “Balanced Itinerary Planning for Multiple Mobile Agents in Wireless Sensor Networks”. Proc of the 2nd Int’l Conf on Ad Hoc Networks (ADHOCNETS’2010), pp. 416-428, 2010.

[9] M. Chen, S. Gonzalez and V.C.M. Leung, “Directional source grouping for multi-agent itinerary planning in wireless sensor networks”. Proc of the 2010 Int’l Conf on Information and Communication Technology Convergence (ICTC’2010), pp. 207-212, 2010.

[10] D. Gavalas, A. Mpitziopoulos, G. Pantziou and C. Konstantopoulos, “An Approach for Near-Optimal Distributed Data Fusion in Wireless Sensor Networks”, Wireless Networks, 16(5), pp. 1407-1425, 2010.

[11] C. Konstantopoulos, A. Mpitziopoulos, D. Gavalas and G. Pantziou, “Effective Determination of Mobile Agent Itineraries for Data Aggregation on Sensor Networks”. IEEE Trans. on Knowledge and Data Eng., 22(12), pp. 1679-1693, 2010.

[12] L. Esau and K. Williams, “On Teleprocessing System Design, Part II – A Method for Approximating the Optimal Network,” IBM Systems Journal, 5(3), pp. 142-147, 1966.

[13] The Castalia simulator for Wireless Sensor Networks: http://castalia.research.nicta.com.