Exploiting the Cloning Capability of Mobile Agents for Cost-Effective Data Fusion in Wireless Sensor Networks

Aristides Mpitziopoulos [a,*], Damianos Gavalas [a]

Charalampos Konstantopoulos [b] and Grammati Pantziou [c]

[a] *Department of Cultural Technology and Communication, University of the Aegean, Lesvos, Greece*

[b] *Research Academic Computer Technology Institute, Patras, Greece*

[c] *Department of Informatics, Technological Educational Institution of Athens, Athens, Greece*

Abstract

Mobile Agent (MA) technology has been recently proposed in Wireless Sensors Networks (WSNs) literature to answer the scalability problem of client/server model in data fusion applications. Herein, we present CBID (Clone-Based Itinerary Design), a novel algorithm that calculates near-optimal routes for MAs that incrementally fuse the data as they visit the nodes while enabling a fast update of designed itineraries which is especially important for highly dynamic topologies. The order of visited nodes highly affects the quality and the overall cost of data fusion. The key MAs characteristic exploited by CBID to reduce energy consumption and response time is their cloning capability. Parallel dispatched MAs sequentially visit sensor nodes arranged in tree structures and upon visiting a node with two or more child nodes, the MAs (master MAs) clone of themselves with each clone (slave MA) visiting a tree branch. When all slave MAs return to that node, report their data to the master MA and are self-destroyed. This results in highly reduced energy consumption and response time since many MAs work in parallel. Simulation results prove the high effectiveness of CBID in data fusion tasks compared to other alternative algorithms.

Key words: Wireless Sensor Networks, mobile agents, jamming avoidance, routing, data fusion, itineraries

* Corresponding author, tel.:+306936028286.

Email addresses: crmaris@aegean.gr (Aristides Mpitziopoulos), dgavalas@aegean.gr (Damianos Gavalas), konstant@cti.gr (Charalampos Konstantopoulos), pantziou@teiath.gr (Grammati Pantziou).

1 Introduction

Data fusion is the process of combining data and knowledge from different sources with the aim of maximizing the useful information content. It improves reliability while offering the opportunity to minimize the data retained. Multiple sensor data fusion is an evolving technology, concerning the problem of how to fuse data from multiple sensors in order to make a more accurate estimation of the environment[10]. Applications of data fusion cross a wide spectrum, including environment monitoring, automatic target detection and tracking, battlefield surveillance, remote sensing, global awareness, etc [1]. They are usually time-critical, cover a large geographical area and require reliable delivery of accurate information for their completion. Most energy-efficient proposals are based on the traditional client/server computing model to handle multisensor data fusion in Wireless Sensor Networks (WSNs); in that model, each sensor sends its sensory data to a back-end processing element (PE) or sink. However, as advances in sensor technology and computer networking allow the deployment of large amount of smaller and cheaper sensors, huge volumes of data need to be processed in real-time.

Recent research works suggested the use of Mobile Agents (MAs) [6], an intrinsically distributed computing technology, in the field of WSNs. The term Mobile Agent (MA) is referred to an autonomous application program able of migrating from node to node to complete specific tasks assigned from network users. A MA can be programmed to determine the local process of data in each sensor node according to the data it already carries and hence the remaining data after each local data fusion. These data are then carried by the MA to the next sensor node where the same procedure is applied.

When MAs are employed for data fusion tasks in WSNs the choice of agents' itineraries (order of visited sensors) is of immense importance because it greatly affects the overall energy consumption and data fusion cost. The most notable drawback of existing solutions that have been proposed in the literature for determining the routes of the MAs [9,13] is that they rely on a single MA to visit and fuse the data from sensor node and this is a serious problem in large scale WSNs where thousands of sensor nodes should be visited by the MA.

The main objective of our proposed CBID algorithm is to calculate near-optimal routes for MAs that incrementally fuse the data as they visit the nodes. An inviting side-effect of CBID is that it enables a fast update of designed itineraries which is especially important for highly dynamic topologies. For instance, the fast execution of CBID makes it suitable for responding to

jamming/interference [7] and also in applications that include tracking and monitoring of moving objects. The main objective is addressed through the design of CBID that not only determines the proper number of MAs for minimizing the total data fusion cost but also constructs low cost itineraries for each of them. The remainder of the paper is organized as follows: Section 2 reviews works related to our research. Section 3 discusses the design and functionality of our heuristic algorithm for designing near-optimal itineraries for mobile agents performing data fusion and security tasks in WSNs. Simulation results are presented in Section 4, while Section 5 concludes the paper and presents future directions of our work.

2 Related work

WSNs pose new challenges as the wireless bandwidth is typically much lower than that of a wired network and sensory data traffic may even exceed network capacity. To solve the problem of the overwhelming data traffic Qi at al in [9] and [10] proposed the MA-based Distributed Sensor Network (MADSN) for scalable and energy-efficient data aggregation. By transmitting the software code (MA) to sensor nodes, a large amount of sensory data may be filtered at the source by eliminating the redundancy. MA may visit a number of sensors and progressively fuse retrieved sensory data, prior to returning back to the PE to deliver the data. This scheme may be more efficient than traditional client/server model; within the latter model, raw sensory data are transmitted to the PE where data fusion takes place (see Fig. 1). In [11,15] the higher performance of the MASDN over the client/server model is demonstrated through both analytical study and simulations. The shortcoming of these studies is that both assume a constant size for the MA, which is not realistic since MAs grow larger as they collect data from distributed nodes.

Xu at al. in [14] study the problem of determining mobile agent itinerary for collaborative processing and models the Dynamic Mobile Agent Planning (DMAP) problem. They present two itinerary planning algorithms, ISMAP and IDMAP. The main drawback of both algorithms is that they rely on a single MA. This causes a problem in large scale WSNs where thousands of sensor nodes should be visited by a single MA.

Qi and Wang in [9] proposed two heuristics to optimize the itinerary of MAs performing data fusion tasks. In 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 Global Closest First (GCF) algorithm, MAs also start their itinerary from the PE node and select the node

closest to the centre of the surveillance region as the next-hop destination.

The output of LCF-like algorithms though highly depends on the MA original location, while the nodes left to be visited last are typically associated with high migration cost [3] 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. On the other hand, GCF produces in most cases messier routes than LCF and repetitive MA oscillations around the region centre, resulting in long route paths and unacceptably poor performance [11,13]. Wu et al. proposed a genetic algorithm-based solution for computing routes for an MA that incrementally fuses the data as it visits the nodes in a WSN [13]. Although providing superior performance (lower cost) than LCF and GCF algorithms, this approach implies a time-expensive optimal itinerary calculation (genetic algorithms typically start their execution with a random solution 'vector' which is improved as the execution progresses), which is unacceptable for time-critical applications, e.g. in target detection and tracking.

Most importantly, both the approaches proposed in [11] and [13] involve the use of a single MA object launched from the PE that sequentially visits all sensors, regardless of their physical location on the plane. Their performance is satisfactory for small WSNs; however, it deteriorates as the network size grows and the sensor distributions become more complicated. This is because both the MA's round-trip delay and the overall migration cost increases exponentially with network size, as the traveling MA accumulates into its state data from visited sensors [2,12]. The growing MA's state size not only results in increased consumption of the limited wireless bandwidth, but also consumes the limited energy supplies of sensor nodes.

To address the above-mentioned problems we proposed the Near Optimal Itinerary Design (NOID) algorithm in [5]. This algorithm adapts a method usually applied in network design problems, namely the Esau-Williams (E-W) heuristic [4], in the specific requirements of sensor networks. Also, not only suggests the optimal number of MAs that minimizes the overall data fusion cost, but also constructs near-optimal itineraries for each of them. To keep the energy consumption low NOID restricts the number of migrations performed by individual MAs, thereby promoting the parallel employment of multiple cooperating MAs, each visiting a subset of sensors. NOID has been shown to perform better than LCF and GCF at the expense of high algorithmic complexity (ie. prolonged execution time). Furthermore NOID does not exploit the cloning capability of mobile agents to reduce the overall data fusion cost even further.

3 The CBID Algorithm

In this section we present the Clone Based Itinerary Design (CBID) algorithm for determining the number of MAs that should be employed and scheduling their corresponding near-optimal itineraries in WSN environments. CBID's execution comprises of three phases. In the first phase (initialization phase) we 'connect' the PE with all sensor nodes located within the PE's transmission range (nodes h and e illustrated in Fig. 1). These nodes represent the starting point of the corresponding MA itineraries; namely, the number of these nodes equals the number of MA itineraries.

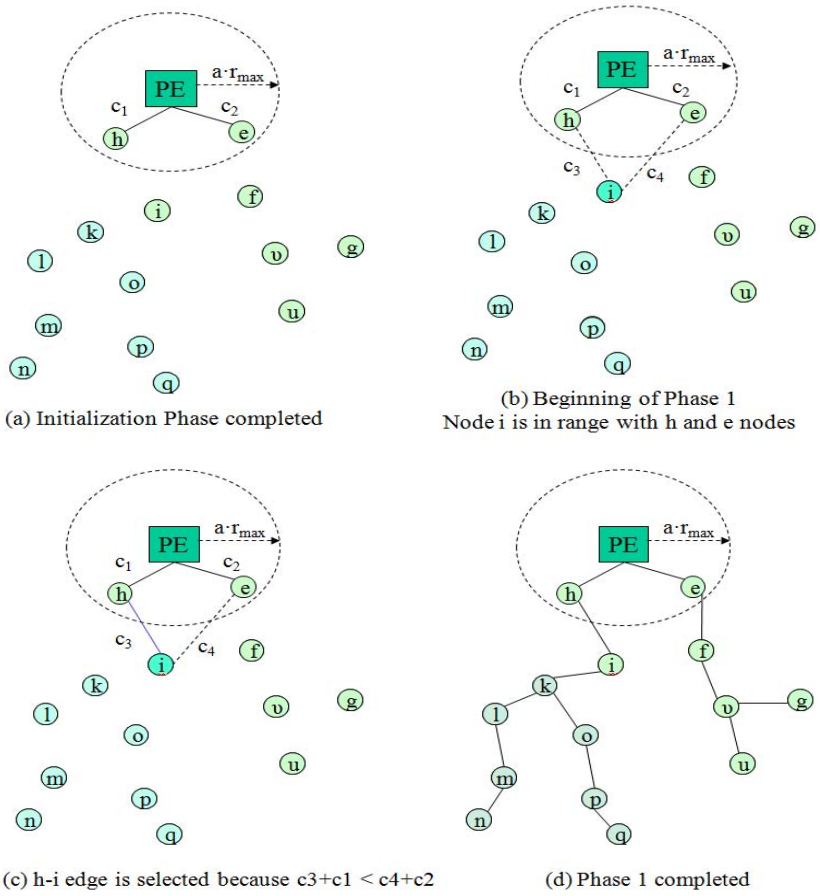


Fig. 1. Itineraries suggested by CBID

To dynamically adjust the number of itineraries, we use the parameter $a \times r_{max}$ where a is an input parameter in the range (0, 1] and r_{max} is the maximum transmission range of the PE. In the second phase (phase 1 illustrated in Fig. 1b,c,d), we attach new nodes to the initially formed trees, so as to maintain low itinerary cost post the attachment. The connection cost is calculated with an efficient way, as defined in definition 1 and exemplified in equation (1). The three basic rules that should be taken into account to establish each connection

are: (a) the candidate for attachment node must not be already attached to another tree; (b) the candidate node is attached to an in-range node that provides a connecting path back to the PE; (c) the node to be connected must keep the itinerary cost to minimum compared to other candidate nodes.

Let as now assume that we have a WSN with the PE connected to a tree with 4 nodes (nodes e,f,υ,g in Fig. 2a,b). The more distant from the PE node is υ and we assume an unconnected node u that is in range of node υ. We consider the edge (u,υ) that connect node u with node υ. We also provide the following definition:

Definition 1 *The Connection Cost CC of edge (u,υ) is the ensuing itinerary cost assuming only one MA follows the itinerary after connecting node u to the tree of node υ provided that υ is in some tree. Otherwise, we have $CC_{(u,\upsilon)} = \infty$.*

$$\begin{aligned} CC_{(u,\upsilon)} &= S \cdot (C_{PE,e} + C_{e,f} + C_{f,\upsilon} + C_{\upsilon,u}) \\ &+ (df + S) \cdot C_{u,\upsilon} + (2df + S) \cdot C_{\upsilon,g} + (3df + S) \cdot (C_{g,\upsilon} + C_{\upsilon,f}) \\ &+ (4df + S) \cdot C_{f,e} + (5df + S) \cdot C_{e,PE} \end{aligned} \tag{1}$$

In equation 1 the individual costs are calculated taking into account only the spatial distance between nodes (although additional parameters such as residual energy could be easily included). S represents the MA initial size in bytes and df equals to the data the MA is carrying (d in bytes) multiplied with data fusion coefficient (f).

With the use of equation 1 we choose the nodes to be connected in phases 1 and 2 among all candidate nodes. We connect the nodes that give the smallest Connection Cost after connection(see Fig. 1c).

The end result is multiple trees, all rooted at the PE. The nodes of each individual tree will comprise the itinerary of separate MAs. Each MA (master MA) starts its itinerary from the PE following its assigned tree structure. When it reaches a 'parent' node with two or more child-nodes (e.g. node k in Fig. 2a) the cloning is performed and each clone (slave MA) is assigned to visit each child-node branch while the master MA remains to the 'parent' node.

When the slave MAs reach the more distant from the PE nodes (n and q nodes in Fig. 2b), they start the collection of data. On the way back to the route, the slave MAs that return to the parent' node deliver their collected data to the master MA and are subsequently self-destroyed. The master MA waits for

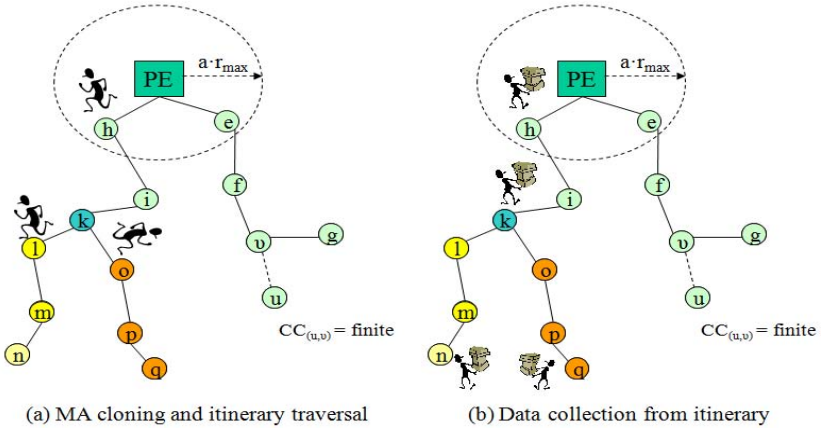


Fig. 2. CBID MA cloning and data collection

a fixed time period for the arrival of all slave MAs. If this time period elapses with no arrival of all slave MAs, it assumes a node failure or communication disconnection and returns back to the PE following the same route. It should be emphasized that the above-described procedure is recursive, that is if a slave MA, on its way to the more distant leaf node, visits another parent' node, it changes its status to Master MA and the same process is repeated.

Using the above-mentioned method the overall data fusion cost and response time (MA round trip delay) are considerably reduced. This is because CBID enables the parallel employment of multiple MAs carrying small loads of data instead of a single MA traveling longer distances and growing heavy, especially in the last hops of its itinerary[2].

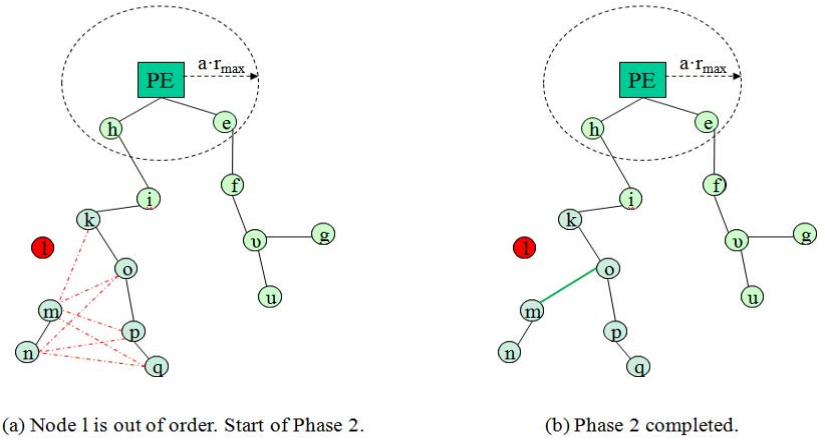


Fig. 3. Itineraries suggested by CBID

Phase 2 is executed only for itinerary updates performed upon topology changes. In this phase we assume that the PE sends on specific time intervals queries to all sensor nodes (e.g. energy level) and logs possible topology changes (e.g. not responding sensor nodes or sensors that report low energy level). All the edges

starting or ending at problematic nodes (node l in Fig. 3a) are deleted. Then nodes with limited connectivity (part of cut-off trees) are identified (nodes m and n in Fig. 3a). All the cut-off trees nodes having in range working nodes are examined (see Fig. 3a) and the corresponding CC values are updated. The nodes with the lowest CC value are then connected (nodes m and o in Fig. 3b).

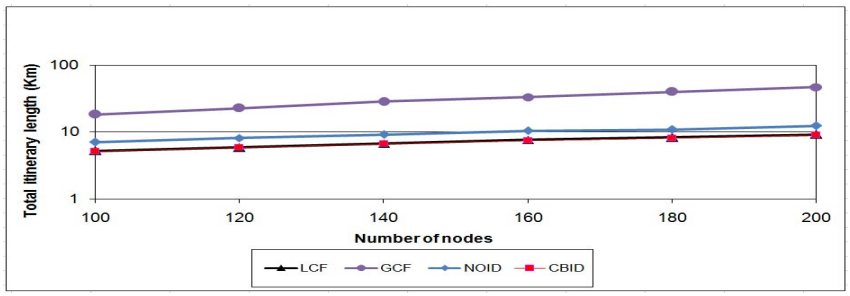
4 Simulation Results

Our simulations tests compare the performance of CBID against NOID, LCF and GCF algorithms in terms of the overall itinerary length, data fusion cost and data fusion response time. Unless otherwise specified, the parameters used throughout the simulation tests are those shown in Table 1. The simulation results presented herein have been averaged over ten simulation runs (i.e. ten different network topologies).

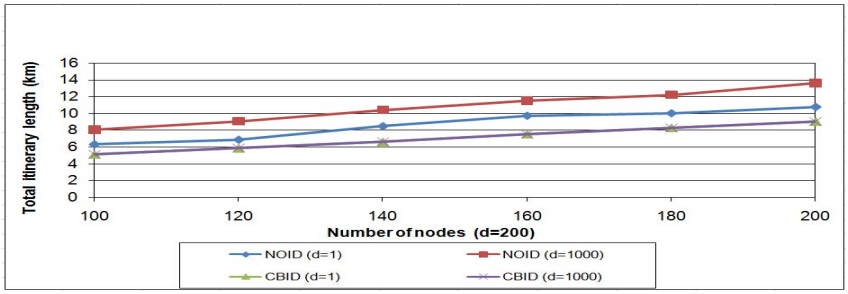
Table 1
Simulation Parameters

Parameter	Value
Simulated plane (m^2)	700×450
# Sensors	100
Sensors transmission power (dBm)	4
Sensors transmission range (m),assuming clear terrain	10
Network transfer rate (Kbps)	250
Initial sensors battery lifetime	100 energy units
MA execution time at each sensor (processing delay, in msec)	50
MA instantiation delay (msec)	10
MA code size (s, in bytes)	1000
Bytes accumulated by the MA at each sensor (d, in bytes)	200
Parameter a (CBID)	1
Data fusion coefficient (f)	1

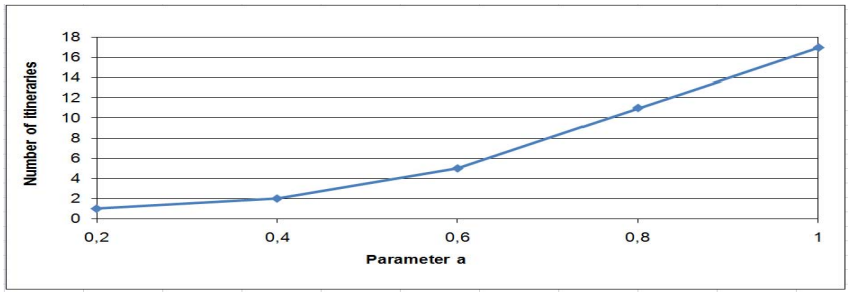
Simulations have been conducted using a Delphi-based tool, implemented for this purpose. In our simulations, without loss of generality, we assume the following parameter:(a) MA code size $s = 1000$ bytes (b) bytes accumulated by the MA at each sensor $d = 200$ bytes (c) parameter $a = 1$ (for CBID), (d) Data fusion coefficient f=1. The simulation results presented herein have been averaged over ten simulation runs (i.e. ten different network topologies. The full paper will include additional results (performance comparison in terms of response time, energy consumption, etc).



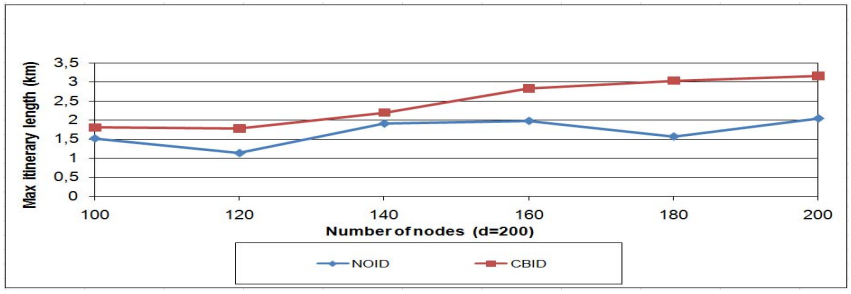
(a)



(b)



(c)



(d)

Fig. 4. Delphi-based simulation of MA-based distributed data fusion algorithms: (a) Total itinerary length of LCF, GCF, NOID and CBID; (b) Total itinerary length of NOID and CBID (d=1 and d=1000); (c) number of itineraries suggested by CBID with parameter a values (0.2-1), BS range=100m, #sensors=100; (d) Max suggested itinerary length of NOID and CBID s=1000 bytes, d=200, a=1 (for CBID)

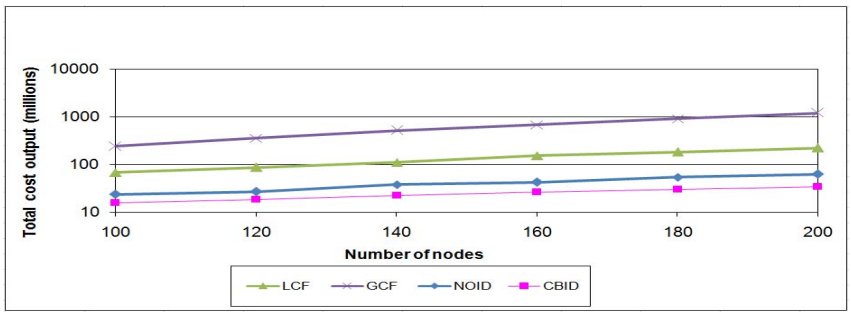
A first set of simulation experiments compares the performance of LCF, GCF, NOID and CBID algorithms in terms of their total itinerary length. As illustrated in Fig. 4a, CBID and LCF have the smaller itinerary length, followed

by NOID. GCF has by far the bigger itinerary length because it suggests wireless hops among distant sensor nodes. In Fig. 4b we can see that d/s ratio appears to have no affect in the total CBID itinerary length, in contrast to NOID wherein as d/s ratio increases, the total length of NOID itineraries increases remarkably (larger number of MAs cooperate in the data fusion task). The number of itineraries in CBID depends on the parameter a and the sink's transmission range (Fig. 4c). Finally Fig. 4d illustrates the total length of the longer suggested itinerary of NOID and CBID in simulated WSNs with variable number of sensors.

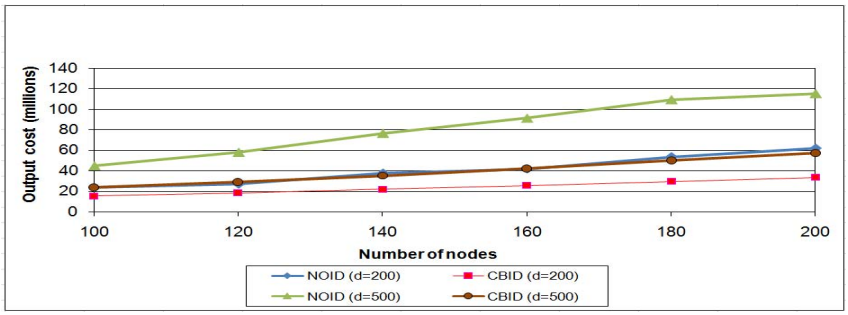
Fig. 5a (drawn on logarithmic scale) compares LCF, GCF, NOID and CBID algorithm in terms of their respective overall data fusion cost. GCF is shown to suggest the larger overall cost, followed by LCF. NOID performs better than LCF and GCF, while CBID is shown to provide significantly lower data fusion cost than NOID as: (a) agent itineraries are designed paying 'respect' to the tree structures (i.e. MAs migrations follow the tree edges, unlike NOID which designs similar tree structures but then constructs agent routes that correspond to a post-order traversal of these trees), (b) NOID fails to exploit the cloning capability of MAs; on the contrary, CBID enables agent cloning whenever this can reduce the fusion cost.

In Fig. 5b we compare NOID and CBID for d=200, d=500 for networks up to 200 sensors. As d and the number of sensors increase, the cost output difference increases in favor of CBID. Fig. 5c verifies that for a given network size the cost benefit of CBID over NOID increases drastically with d/s ratio. As d increases the output cost of CBID increases in a slower rate, mainly due to the cloning feature of MAs that CBID takes advantage off. Furthermore as explained in section 3, in CBID the MAs (and their clones) visit first the more distant leaf nodes of the itinerary and then on their way to the PE start the collection of data from the other sensors. On the other hand, in NOID, MAs commence data collection from the node closest to the PE and as soon as they reach the more distant itinerary nodes, they travel all the distance back to the Sink carrying the data retrieved from all visited sensors.

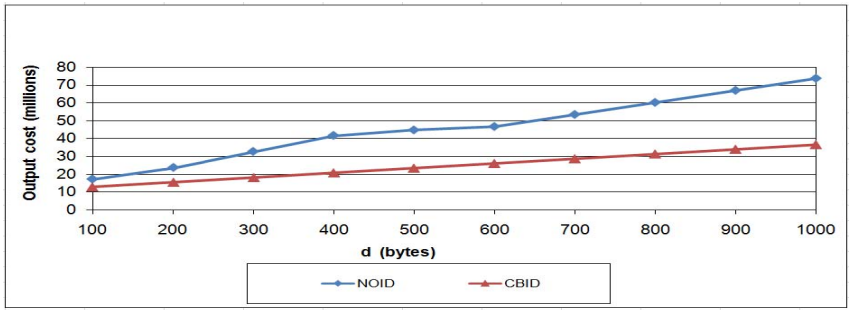
Finally, Fig. 5d compares the total cost output of CBID and NOID with variable number of nodes in the area (we set a=1 because NOID does not support the a parameter) from 20% to 100% (in the latter case, all sensor nodes are located within the area). From the results we can see that CBID shows an increase in cost output as the number of nodes in the area. In contrary the output cost of NOID decreases since this variable does not affect its output.



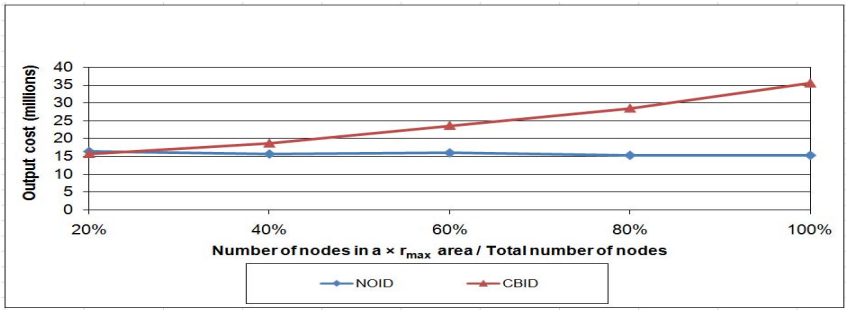
(a)



(b)



(c)



(d)

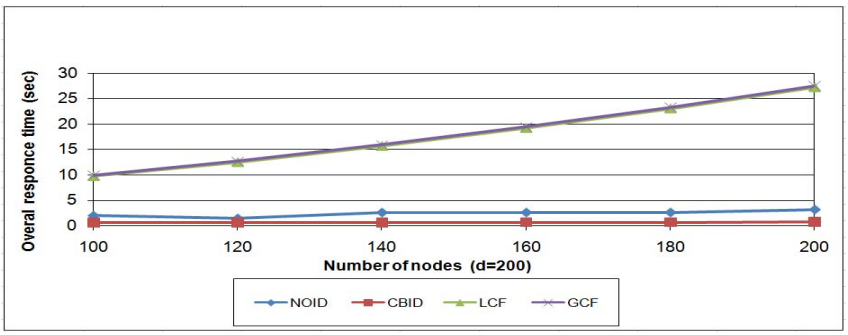
Fig. 5. Comparison of LCF, GCF, NOID and CBID in data fusion cost for (a) s=1000 d=200 bytes; (b) comparison of CBID, NOID s=1000, d=200 bytes, d=500 bytes; (c) s=1000 bytes, network size of 200 sensors; (d) s=1000 bytes, d=200 bytes, 20-100% of overall number of nodes in $a \times r_{max}$ area

The next set of experiments evaluates the overall response time of LCF, GCF, NOID and CBID algorithms for completing data fusion tasks. Response time is calculated as the sum of MAs instantiation delay, processing delay, MAs transmission delay and propagation delay:

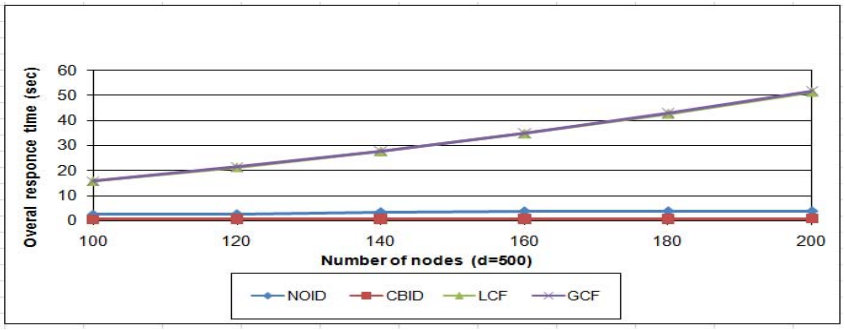
$$t_{overall} = t_{inst} + t_{proc} + t_{trans} + t_{prop} \quad (2)$$

The MAs instantiation delay (t_{inst}) is related to the number of MAs involved in the data fusion task (in our experiments it takes 10 msec to instantiate each MA object). Hence, it is constant for LCF and GCF algorithms that always instantiate a single MA that visits the whole set of sensor nodes, while for NOID it depends on the network scale and the d/s ratio which dictates the number of proposed itineraries. In CBID it depends also on the network scale taking into account parameter a (see Fig. 4c) and the communication range of the Sink. The processing delay (t_{proc}, time needed for the MA to complete its data fusion task on each sensor) is constant (50 msec in our experiments). The transmission delay (t_{trans}) depends on the network transfer rate and the current size of the MA (i.e. the MA's code size plus the amount of data accumulated within the MA's state). Finally, the propagation delay (t_{prop}) depends on the physical distance covered in successive MA migrations (i.e. on the overall itinerary length).

Response time measurements are depicted in Fig. 6a and Fig. 6b. In both graphs, the response times of LCF and GCF almost coincide: LCF only involves slightly decreased propagation delay compared to GCF since it derives shorter itinerary lengths (see Fig. 4a). It is demonstrated that as the d/s ratio and the number of nodes increases (see Fig. 4b) the response time gain of NOID and CBID over LCF and GCF increases drastically as the transmission time dominates (for LCF and GCF) over the other delay parameters. That is, although NOID and CBID dispatches a large number of MAs thereby increasing t_{inst}, these MAs travel in parallel, each visiting a small set of sensors (unlike LCF and GCF where a single MA performs a number of hops equal to the number of sensors). Hence, in NOID and CBID cases, by the end of their itinerary MAs have not collected large chunks of data, considerably decreasing the associated transmission delay. Finally, Fig. 6a and Fig. 6b show that CBID involves smaller overall response time than NOID mostly due to the cloning feature of MAs that utilizes in data collection. As d increases CBID retains the smaller overall response time compared to LCF, GCF and NOID (see Fig. 6b).



(a)



(b)

Fig. 6. Comparison of LCF, GCF, NOID and CBID algorithms in terms of the overall response time for (a) d=200 bytes, s=1000 bytes, (b) d=500 bytes, s=1000 bytes.

The conclusions of our simulation results indicate that CBID:

- achieves smaller overall data fusion cost (thus lower energy consumption for the nodes).
- retains the smaller overall response time compared to NOID.
- in case(s) of possible topology change(s) in a WSN is able to update the itineraries followed by the MAs very fast.

However, the above advantages are offered at the expense of more complex manageability of the data fusion task. That is, an effective inter-agent communication protocol should be devised to allow exchange of data among the individual clones (e.g. through the 'message board abstraction). In addition, the creation of multiple MA clones may consume valuable nodes resources.

5 Conclusion

In this article we presented CBID, an efficient heuristic algorithm that derives near-optimal itineraries for MAs performing incremental data fusion in WSN environment. CBID is also able to address possible topology changes modifying

MA itineraries through a fast update, so as to exclude problematic nodes from the data fusion process. CBID achieves lower data fusion cost and smaller response time, compared to alternative approaches.

As a future work, we intend to implement CBID in real WSN environments and evaluate its performance in various applications. A set of sensor nodes capable of hosting and providing an execution environment for MAs programmed in Java [8] will form the basis of our field trials.

References

[1] F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, "A Survey on Sensor Networks", IEEE Communications Magazine, pp. 102-114, August 2002.

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

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

[4] L.R. Esau, K.C.Williams, On Teleprocessing System Design, Part II- A Method for Approximating the Optimal Network, IBM Systems Journal, 5, 142-147, 1966.

[5] A. Mpitziopoulos, D. Gavalas, C. Konstantopoulos and G. Pantziou, "Deriving Efficient Mobile Agent Routes in Wireless Sensor Networks with NOID Algorithm", Proceedings of the 18th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'2007), September 2007.

[6] V.Pham,A.Karmouch,"Mobile Software Agents: An Overview", IEEE Communications Magazine, 36(7), pp. 26-37, 1998.

[7] E. Shi, A. Perrig, "Designing Secure Sensor Networks", Wireless Communications Magazine, 11(6), pp. 38-43, December 2004.

[8] Sun Microsystems, Sun Spot project home page, http://www.sunspotworld.com/.

[9] H. Qi, F. Wang, "Optimal Itinerary Analysis for Mobile Agents in Ad Hoc Wireless Sensor Networks", Proceedings of the 13th International Conference on Wireless Communications (Wireless'2001), pp.147-153, 2001.

[10] H. Qi, S.S. Iyengar, K. Chakrabarty, "Multi-Resolution Data Integration Using Mobile Agents in Distributed Sensor Networks", IEEE Transactions on Systems, Man, and Cybernetics, Part C, 31(3), pp. 383-391, August 2001.

[11] H. Qi, Y. Xu, X. Wang, "Mobile-Agent-Based Collaborative Signal and Information Processing in Sensor Networks", Proceedings of the IEEE, 91(8), pp. 1172-1182, 2003.

[12] M.G. Rubinstein, O. C. Duarte, G. Pujolle, "Scalability of a Mobile Agents Based Network Management Application", Journal of Communications and Networks, 5(3), September 2003.

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

[14] Y.Xu, H. Qi, "Dynamic mobile agent migration in Wireless Sensor Networks", International Journal of Ad Hoc and Ubiquitous Computing (IJAHUC), Vol. 2, No. 1/2, 2007

[15] Y. Xu, H. Qi, "Distributed Computing Paradigms for Collaborative Signal and Information Processing in Sensor Networks", Journal of Parallel and Distributed Computing, 64, 945-959, 2004.