# Mobility support for wireless sensor networks

Filippe C. Jabour
UFRJ - PEE/COPPE/GTA
CEFET-MG - Brasil
jabour@ieee.org

Eugênia Giancoli
UFRJ - PEE/COPPE/GTA
CEFET-MG - Brasil
eugenia@gta.ufrj.br

Aloysio C. P. Pedroza
UFRJ - PEE/COPPE/GTA
Brasil
aloysio@gta.ufrj.br

## Abstract

*This work proposes a two-tier approach for mobility support in wireless sensor networks. It is based on local interactions among sensors, on global tasks of mobile agents and on location prediction. We demonstrate the correctness of a simple location prediction model. We also propose, evaluate and compare two algorithms for mobile agents decision. The proposed scheme is stateless and does not need a routing protocol. All computing (location prediction and mobile agents decision) are of linear complexity. We observed a better performance as mobility degree and node density grow.*

## 1 Introduction

A wireless sensor network (WSN) is a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations [13, 9].

In dynamic sensor networks, either the sensors themselves, the observer, or the phenomenon are mobile. Whenever any of the sensors associated with the current path from the observer to the phenomenon moves, the path may fail. In this case, either the observer or the concerned sensor must take the initiative to rebuild a new path [16]. Sensor nodes mobility can result from environmental influences such as wind or water, sensors may be attached to or carried by mobile entities, or they may possess automotive capabilities [13]. Some scenarios with mobility: sensors used to study tornado movements [16]; sensors connected to vehicles in big cities to study traffic conditions and routing planning [16]; sensors floating on rivers currents [10]; NASA's project to exploration of Mars, with sensors driven by winds [3].

This work proposes a two-tier approach for mobility support in WSN. It is based on local interactions among sensors, on global tasks of mobile agents (MA) and on location prediction. We present and evaluate a simple algorithm for node location prediction. We also propose, evaluate and compare two algorithms for MAs decision process.

In section 2 we discuss some related works. Section 3 presents the location prediction algorithm. In section 4 we propose two decision algorithms used by MAs to perform its tasks. Performance evaluation and algorithms comparison are in section 5. In section 6 we conclude and present the future works.

## 2 Related works

In mobile scenarios, we cannot assume the presence of a connected path from source to destination. Epidemic Routing [18] presents techniques to deliver messages in the case where there is never a connected path from source to destination or when a network partition exists at the time a message is originated. Random pair-wise exchanges of messages among mobile hosts ensure eventual message delivery. Our work also explores the mobility to find a path towards the destination host. However, while Epidemic Routing distributes copies of the same message by the network, our work does not maintain any state inside the network (see section 4).

Choksi et al. [6] present four enhancements to directed diffusion [11] to support mobile users. The first is a small change to diffusion which restarts path discovery when a node moves. The second is a simple handoff scheme similar to those used in cellular networks. The third introduces special proxy nodes as path anchors, and the fourth scheme provides hints to the diffusion layer to pre-create routes in an anticipated direction of motion. The simulation results show that a combination of these enhancements to diffusion permit sink speeds of up $7m/s$ without generating excessive overhead. While this work have simulated only sink mobility, we have investigated general mobility: of all nodes, sources and sinks.

Shah et al. [14, 15] propose an approach that exploits the presence of mobile entities (called MULEs) present in the environment. When in close range, MULEs pick up data

from the sensors, buffer it, and deliver it to wired access points. This can lead to substantial power savings at the sensors as they only have to transmit over a short range. Our work incorporates some techniques of Data MULEs, but they are performed by software entities (MAs), not by "real" entities. Data MULEs move randomly while our MAs move according to geographic defined tasks.

Tong et al. [17] present an architecture called SEnsor Networks with MAs (SENMA). MAs in SENMA are powerful hardware units, both in their communication and processing capability and in their ability to traverse the sensor network. This approach shifts computationally intensive tasks away from primitive sensors and shows a substantial gain in energy efficiency. As these powerful hardware units may be unavailable, we incorporate some functionalities of this work and adapt them to MAs running on homogeneous nodes.

## 3 Node's location prediction

A localization algorithm computes periodically the node's location $P$. With two of these measures $P_i(x_i, y_i)$ and $P_{i-1}(x_{i-1}, y_{i-1})$, the node computes the slope $m$ of the straight line defined by $P_i$ and $P_{i-1}$. In equation 1, $m_i$ represents the node's movement direction; $m_{pred_i}$ (equation 2) is the predicted next movement direction; $m_{pred_{i-1}}$ is the last prediction; and $\alpha$ is used to give more or less weight to the last computed slope $m_i$ in connection with the history of the estimated directions $m_{pred_{i-1}}$. As shown in section 4, these results are sufficient for a correct migration decision of MAs.

$$m_i = \frac{y_i - y_{i-1}}{x_i - x_{i-1}} \quad (1)$$

$$m_{pred_i} = \alpha\, m_i + (1 - \alpha)\, m_{pred_{i-1}} \quad (0 \leq \alpha \leq 1) \quad (2)$$

Once the slope of the movement $m_i$ is estimated, the distance that a node will move in the next step is estimated in the same way (equation 3). The term $d_{pred_{i-1}}$ is the last prediction and $d_i$ is calculated in each step based on node's "real" localization. The "real" localization comes from localization algorithm and incorporates an intrinsic error. The term $\gamma$ has the same function as $\alpha$.

Both predictions computation, $m_{pred_i}$ and $d_{pred_i}$ are of constant complexity.

$$d_{pred_i} = \gamma\, d_i + (1 - \gamma)\, d_{pred_{i-1}} \quad (0 \leq \gamma \leq 1) \quad (3)$$

Figures 1, 2 and 3 show the predictor simulation results for two types of trajectories: random and Manhattan. We used $\alpha = 0.5$ and $\gamma = 0.5$ in both cases. In table 1 we have some statistics of predicted values. We observe a good behavior of the predictor. As shown in the next section, these results are sufficient for a correct migration decision of MAs.

**Table 1. Predictor statistics (meters)**

| | Figure 1 | Figure 2 |
|---|---|---|
| Minimum error | $0,74$ | $0,31$ |
| Maximum error | $11,13$ | $16,25$ |
| Average error | $4,67$ | $6,80$ |
| Error standard deviation | $2,35$ | $3,85$ |

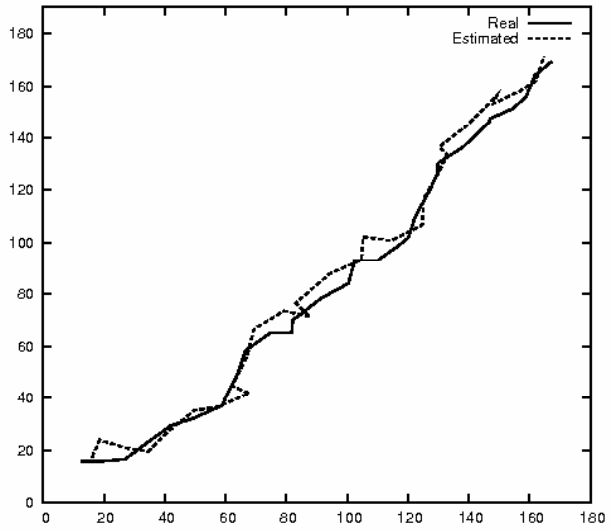


**Figure 1. Location prediction for random trajectory**

## 4 Mobile agents decision algorithms

### 4.1 Local interactions

We focus on scenarios described in section 1, where nodes move randomly by the field of sensing. We consider that the nodes execute an algorithm that allows the knowledge of their location at any given moment, with some error. The nodes estimate their future positions and the slope of their trajectories (section 3).

All nodes maintain a list of reached neighbours at a given moment ($N_i$). It can be done by the *Internet MANET Encapsulation Protocol (IMEP)* [7]. The protocol incorporates mechanisms for supporting link status and neighbour connectivity sensing.

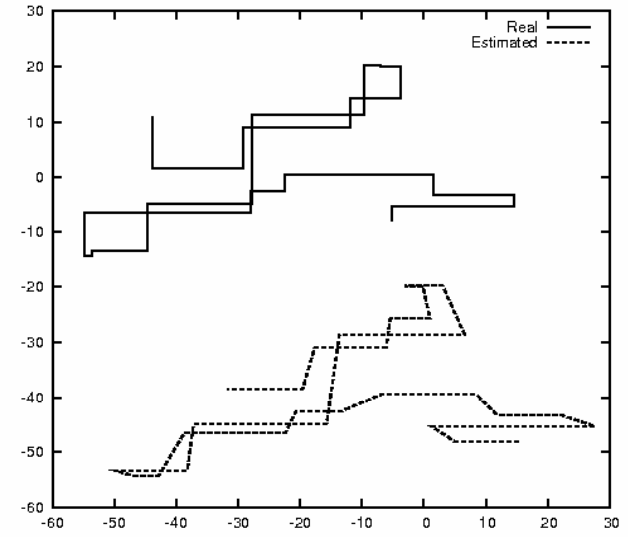


**Figure 2. Location prediction for Manhattan trajectory (y-axis translation for better visualization)**

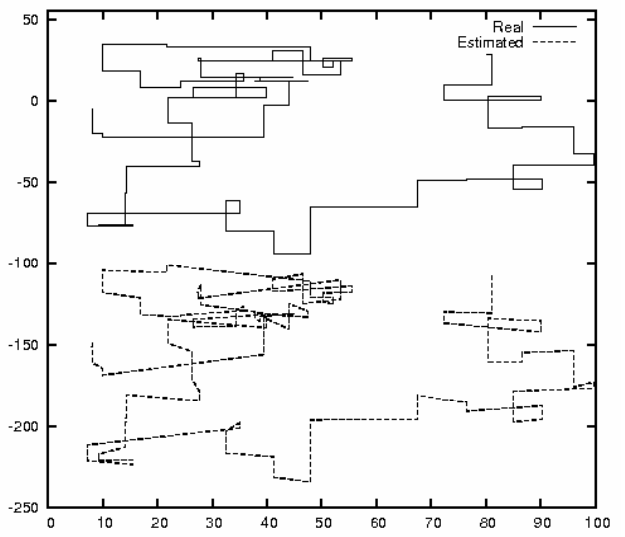


**Figure 3. Location prediction for Manhattan trajectory (y-axis translation for better visualization)**

*Self-organization Protocols for Wireless Sensor Networks* [5] proposes a set of distributed algorithms that provides the basis for a complete self-organized sensor network. The combination of SMACS and Mobile MAC, presented in [5], is suitable for local interactions proposed in our work.

## 4.2 Global tasks

"Generally speaking, MA is a special kind of software which can execute autonomously. Once dispatched, it can migrate from node to node performing data processing autonomously" [12]. Some advantages of using MAs in WSN are scalability, bandwidth saving, reprogramming facilities [12] and data aggregation [4]. Many other MA applications are listed in [8], where an implementation of a MA middleware for MICA2 motes and the development of several applications are discribed and evaluated.

The MAs have sensing tasks to be executed. These tasks contain the coordinates of the region where data should be collected from and the coordinates of the region where data should be brought to. The current destination of a MA is called **target region** and is defined by two points in opposite corners.

When injected into the network, the MA starts an instance of the migration process, searching the destination, towards the target region. Periodically, the MA consults the current position estimated by its host. If it is already inside the target region, it collects the data of interest and starts a new instance of the migration process, towards the return region. If the host is not inside the the target region, the MA executes the migration decision algorithm. We propose two decision algorithms.

### 4.2.1 Distance-only algorithm

The MA identifies in the neighbours set $N_i$ (including its current host), which of them is closer to the target region. If it is the MA's current host, no action is taken. If it is one of its neighbours, the MA starts the migration process to this node.

### 4.2.2 Distance-and-slope algorithm

While performing the location algorithm (section 3), the node calculates a parameter we call **movement type**. This parameter assumes the values $S$(stopped), $U$(up), $D$(down), $R$(right) or $L$(left).

The MA also computes the movement type related to the straight line defined by its host current location and the center of the target region, i.e., the slope of the **optimum trajectory**. We call this value **optimum movement type**. The slope of the optimum trajectory is called the **optimum angular coeficient**. If there is one or more nodes with the same movement type as the optimum movement type, the MA identifies the node with the trajectory's slope closer to the optimum angular coeficient. If it is the MA's current host, no action is taken. If it is one of its neighbours, the MA starts the migration process to this node. If all nodes in $N_i$, including the current host, have different movement

types from optimum movement type, the "distance-only" algorithm is used.

No routing algorithm is needed and no state needs to be stored in the network. Both algorithms are of linear complexity: *O(k)*, where $k$ is the number of neighbours and $k < n$ ($n$ is the number of nodes of the network).

## 5 Performance evaluation and algorithms comparison

We built a discrete event simulator to validate and compare the proposed algorithms. It is a Java [1] application that reads an *ns-2* [2] formatted file containning the mobility instructions for every node of the WSN. The nodes are all configured with the same communication radio range (in meters) and the localization algorithm error (in percentage). The simulator is configured with a clock precision from zero to twelve decimal places. Only when all threads have finished the computation of one step, the clock moves to the next one, according to the clock precision. Each mobile node, each MA and the simulation clock are executed by one independent and concurrent thread. Finally, some MAs are injected into the network with some parameters: initial host, initial processing instant, the destination region coordinates, the return region coordinates and the decision algorithm to be used.

A localization algorithm is assumed to be running and a new mesured location is generated for each node, at each step of the clock. The neighbours set $N_i$ of each node is considered available and is provided by the simulator. When the MA decides to migrate from its host to some neighbour, it is done in one step of the clock, with 100% of success probability.

The mobility scenarios were created with the *setdest* program that commes with *ns-2*. For a defined area, this application generates (1) a random initial point and (2) a random destination point for a mobile node. It also generates (3) a random speed to reach this point and (4) a random period of pause before arriving. Then, the process repeats from step (2) to (4) until a defined final instant. We used the following parameters:

• Number of nodes: from 10 to 600, randomly distributed and randomly moved.
• Node speed: normal distribution between 1 and $10m/s$; normal distribution between 1 and $30m/s$.
• Pause interval: uniform distribution between 0 and $20s$.
• Area: $1000m$ x $1000m$.
• Time of simulation: $10000s$.

Other simulation parameters:

• Clock precision (step): 0 decimal places (step = $1s$).
• Node radio range $50m$.
• Localization algorithm error: 0%.
• Destination region: $A_D(900, 1000), B_D(1000, 900)$. $A_D$ is the up-left corner and $B_D$ is the bottom-right corner.
• Return region: $A_R(0, 100), B_R(100, 0)$. $A_R$ is the up-left corner and $B_D$ is the bottom-right corner.
• MAs are randomly injected in some node (one per node, bounded in 300). MAs start at $5s$ and executes until simulation finishes.

Each time a MA reaches the target region, we count **one hit**. Then, the target region turns from destination to return region (or vice versa). The simulator counts the **number of migrations** performed by MAs and register the instant of all events of interest.

From figure 4 we may conclude that both algorithms profit from mobility, i.e., more mobility (more speed) means more hits. As expected, as nodes density grows, both algorithms scale linearly. Finally, we see that the "distance-only" algorithm is the better one, but we are currently investigating situations where the direction of the movement has to be considered.

In figure 5, migrations are related to energy consumption. As expected, more hits mean more migrations and consequently more energy consumption.

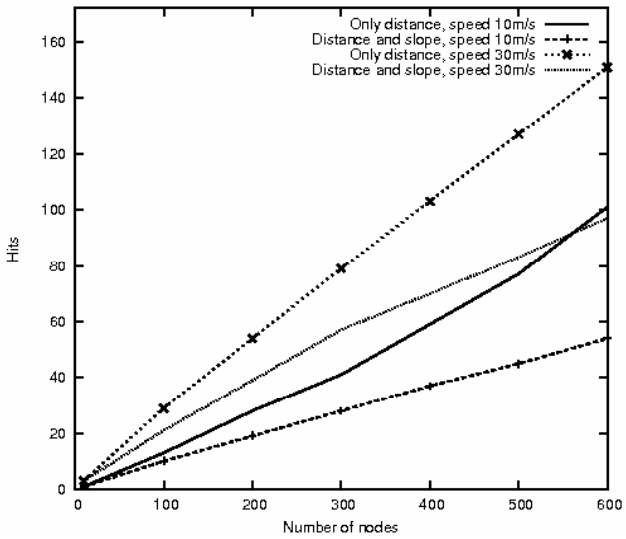


**Figure 4. Number of hits per MA x Number of nodes (10m/s and 30m/s)**

## 6 Conclusions and future works

Since there are few proposals to treat general mobility in WSN, we consider this work a promising approach.

Our simple prediction location model of constant complexity is suitable for the scarce resources of WSNs and the predictions are sufficient for the decision process of MAs.

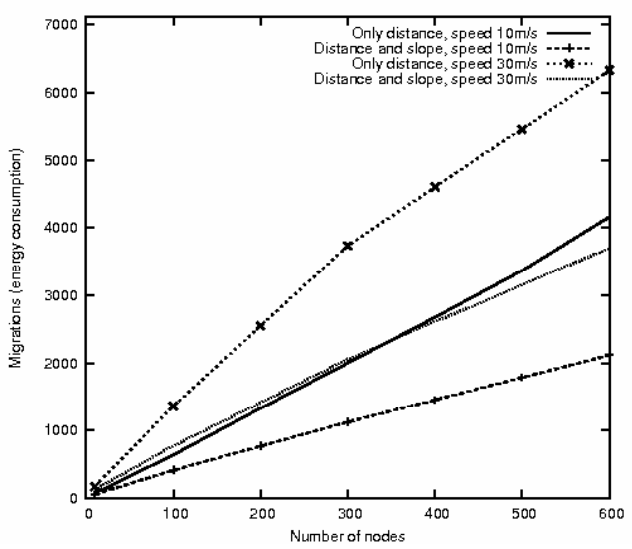


**Figure 5. Total number of migrations per MA x Number of nodes. Related to energy consumption**

The MA mobility algorithms save resources of the network. That's because no routing algorithm is needed and no state is stored in the network. Both algorithms profit from mobility, rather than degrading with it. In the analyzed scenarios, the "distance-only" algorithm was always better than "distance-and-slope". However, we believe that in environments with more mobility (speed) of nodes and less radio range in sensors, this parameter should be essential for the decision process of MAs. These tests are among the experiments that are ongoing.

As future work, we want to embed the error of location and reassess the two algorithms for MAs decision. We have already noticed that, with these errors, the computation of the node's trajectory slope fails. This occurs due to the fact that an error in localization algorithm may occur in any direction. So it may be interpreted, by mistake, as a change in the node's movement direction.

The parameters $\alpha$ and $\gamma$ may be changed according to different mobility scenarios.

In next version of the simulator, we intend to embed stochastic processes to model real transmission losses in communication channels.

## References


[1] Java. http://java.sun.com.

[2] ns-2 network simulator. http://www.isi.edu/nsnam/ns, 1998.

[3] J. Antol, P. Calhoun, J. Flick, G. A. Hajos, R. Kolacinski, D. Minton, R. Owens, and J. Parker. Low cost mars surface exploration: The mars tumbleweed. August 2003. NASA Langley Research Center. NASA/TM-2003-212411.

[4] M. Chen, T. Kwon, Y. Yuan, Y. Choi, and V. C. M. Leung. Mobile agent-based directed diffusion in wireless sensor networks. *EURASIP J. Appl. Signal Process.*, 2007(1):219–219, 2007.

[5] C. Chevallay, R. E. Van, and D. T. A. Hall. Self-organization protocols for wireless sensor networks. In *In Thirty Sixth Conference on Information Sciences and Systems*, 2002.

[6] A. Choksi, R. P. Martin, B. Nath, and R. Pupala. Mobility support for diffusion-based ad-hoc sensor networks. Technical report, Department of Computer Science, Rutgers University, April 2002.

[7] M. S. Corson, S. Papademetriou, P. Papadopoulos, V. Park, and A. Qayyum. An internet MANET encapsulation protocol (IMEP) specification. *INTERNET-DRAFT*, August 1999. IETF MANET Working Group.

[8] C.-L. Fok, G.-C. Roman, and C. Lu. Rapid development and flexible deployment of adaptive wireless sensor network applications. *25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, pages 653–662, 2005.

[9] T. Haenselmann. Wireless sensor network textbook. http://www.informatik.uni-mannheim.de/~haensel/sn_book, April 2006. Retrieved on 2008-08-26.

[10] L. Hu and D. Evans. Localization for mobile sensor networks. *MobiCom '04*, pages 45–57, 2004. ACM.

[11] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 56–67, New York, NY, USA, 2000. ACM Press.

[12] H. Qi, S. S. Iyengar, and K. Chakrabarty. Multi-resolution data integration using mobile agents in distributed sensor networks. In *IEEE-Systems Man Cybernetics*, volume 31, n. 3, pages 383–390, Ago 2001.

[13] K. Römer and F. Mattern. The design space of wireless sensor networks. *IEEE Wireless Communications*, 11(6):54–61, Dec 2004.

[14] R. C. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: Modeling a three-tier architecture for sparce sensor networks. May 2003. IEEE SNPA Workshop, May 2003.

[15] R. C. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: Modeling and analysis of a three-tier architecture for sparse sensor networks. *Ad Hoc Networks*, 1(2-3):215–233, September 2003.

[16] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman. A taxonomy of wireless micro-sensor network models. *ACM SIGMOBILE Mobile Computing and Communications Review*, 6(2):28–36, April 2002.

[17] L. Tong, Q. Zhao, and S. Adireddy. Sensor networks with mobile agents. In *in Proc. 2003 Military Communications Intl Symp*, pages 688–693, Oct 2003. Boston, MA., http://acsp.ece.cornell.edu/.

[18] A. Vahdat and D. Becker. Epidemic routing for partially-connected ad hoc networks. April 2000. Technical Report CS-200006, Duke University, April 2000.