Mobility support for wireless sensor networks

Filippe C. Jabour *•, Eugênia Giancoli *• and Aloysio C. P. Pedroza *

* UFRJ - PEE/COPPE/GTA. • CEFET-MG - Campus Leopoldina. Brazil

email: {jabour,eugenia,aloysio}@gta.ufrj.br

Introduction

This work proposes a two-tier approach for mobility support in wireless sensor networks (WSN). It is based on local interactions among sensors, on global tasks of mobile agents (MA) 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 MA decision) are of linear complexity. We observed a better performance as mobility degree and node density grow.

- WSN: a large number of sensor nodes that cooperate among themselves to monitor an area.
- Mobility: environmental influences such as wind or water; sensors attached to or carried by mobile entities; sensors possess automotive capabilities.
- Mobility has a large impact on the WSN. The speed of movement may also have an impact.
- Whenever any of the sensors associated with the current path from the observer to the phenomenon moves, the path may fail.

Our proposal

- Local interactions: Nodes maintain a present and future estimated localization and a list of neighbours
- Global tasks: MAs play a global role, travelling towards a target region to bring back data.

Location prediction model

A localization algorithm computes periodically the node's location. 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 eq. 1, m_i represents the node's movement direction; m_{pred_i} (eq. 2) is the predicted next movement direction; $m_{pred_{i-1}}$ is the last prediction; and α is used to give more or less weight to the last computed slope m_i in connection with the history of the estimated directions $m_{pred_{i-1}}$.

Fig. 1 and 2 show the predictor simulation results for two types of trajectories ($\alpha = 0.5$). We have concluded that these results are sufficient for a correct migration decision of MAs.

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

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

MA support

- Local interactions maintain a neighbour set and a predicted trajectory for each node.
- MAs are injected into the network with a given task.
- This task must have a **target region** and the data type to be collected.
- When the MA reaches the target region, it performs the same algorithm, searching for the **return region**.
- No routing algorithm is needed, no state must be stored in the network and all computing (location prediction and MA decision) 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).

Two decision algorithms:

1. Based only on distance from the target region. MA migrates to the node that is closer to the target region.
2. Hybrid. MA considers first the direction, migrating to the node where the movement direction is closer to the optimum straight line towards the center of the target region. If no neighbour node is moving towards the target region hemisphere ($180°$), the distance is considered.

Simulations were made to validate and compare the algorithms. We conclude from fig. 3 that both algorithms scale with number of nodes and speed growth, i.e., we have more hits. We also observe that the "only distance" algorithm is the better one, but we are currently investigating situations where the direction of the movement has to be considered. Fig. 4 shows the number of migrations of MAs. Migrations are related to energy consumption. As expected, more hits mean more migrations and consequently more energy consumption.

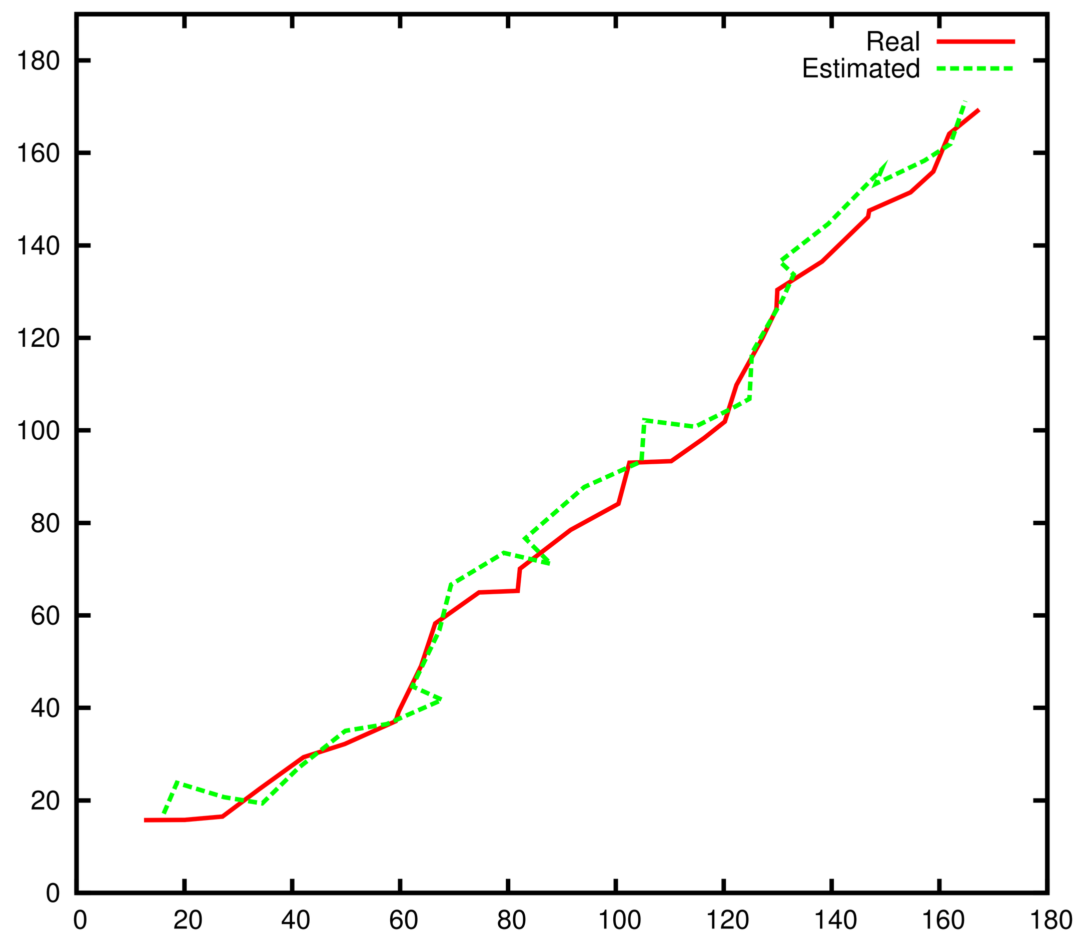

Figure 1: Location prediction; random trajectory

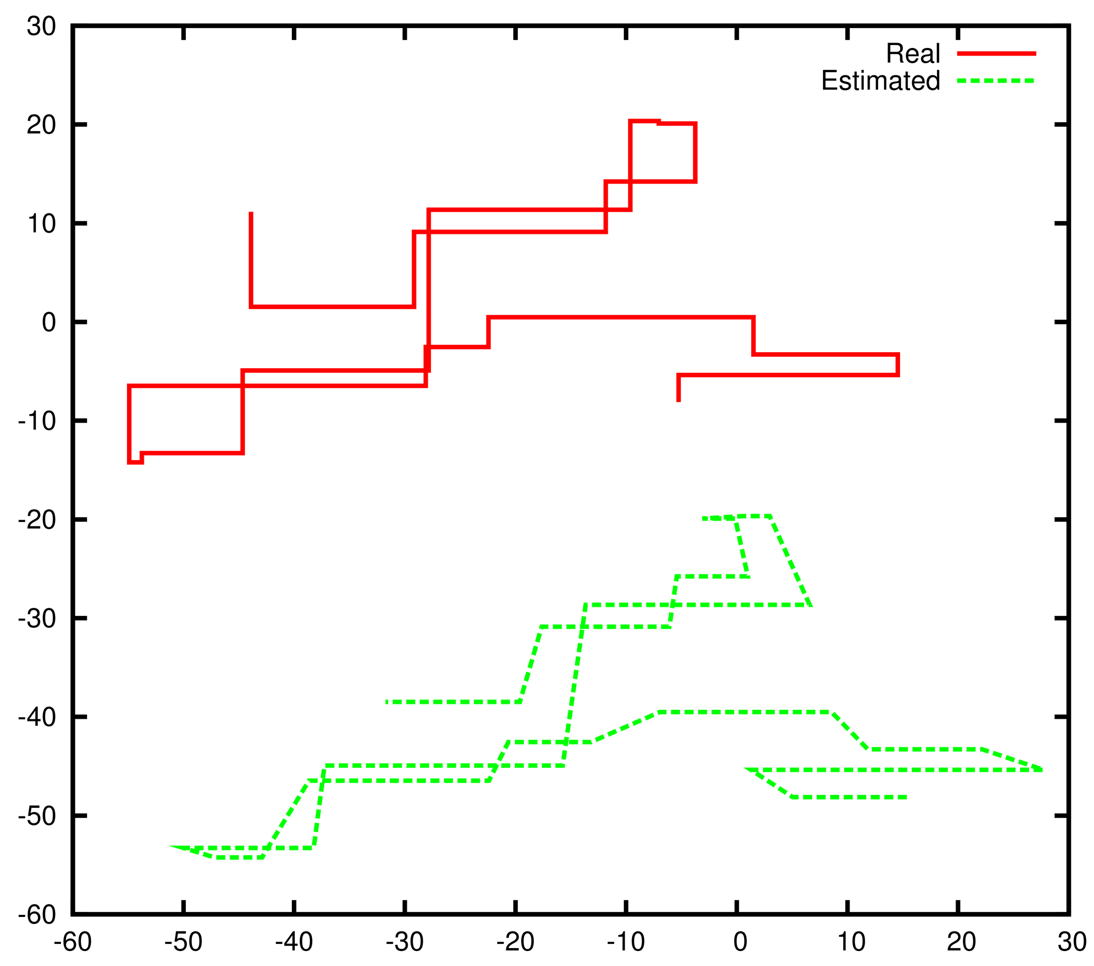

Figure 2: Location prediction; Manhattan trajectory.

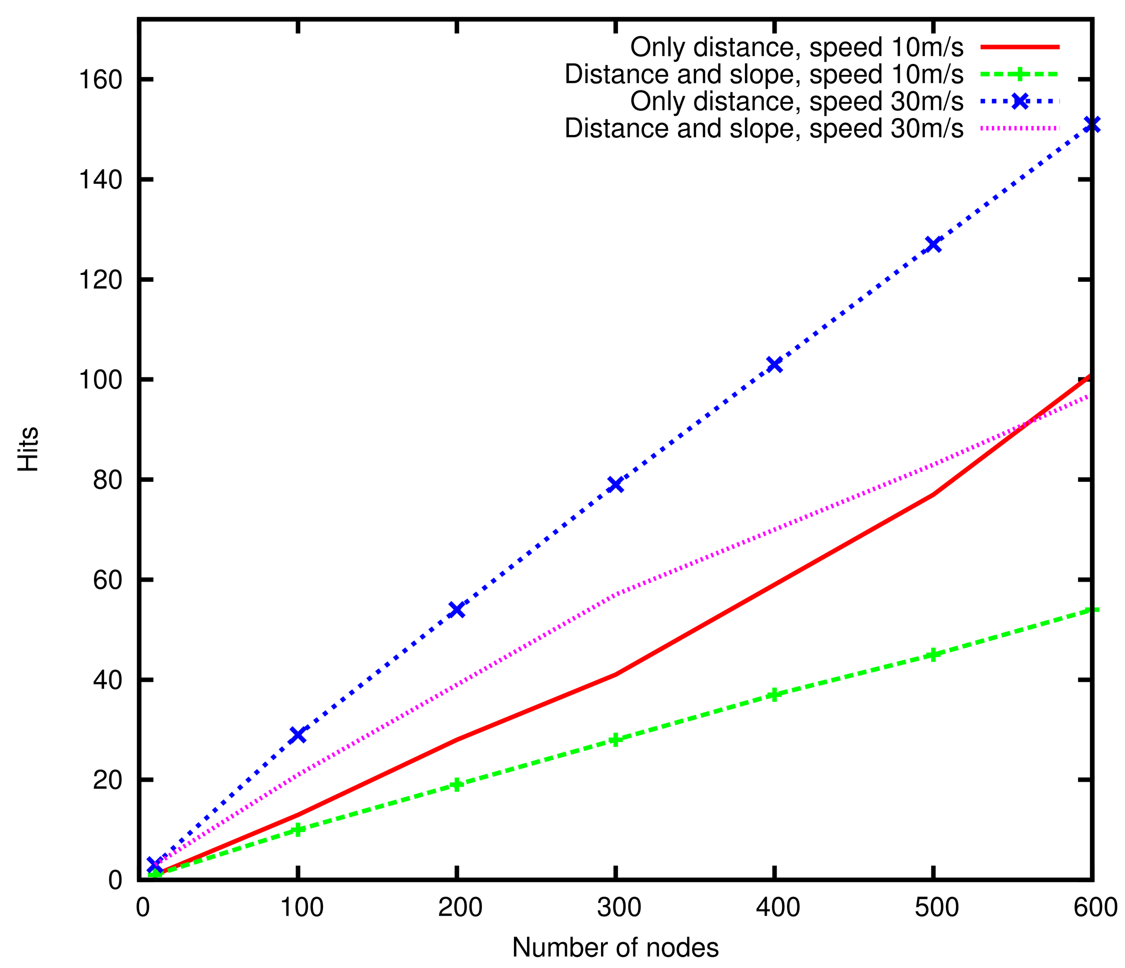

Figure 3: MA simulation results. Hit means

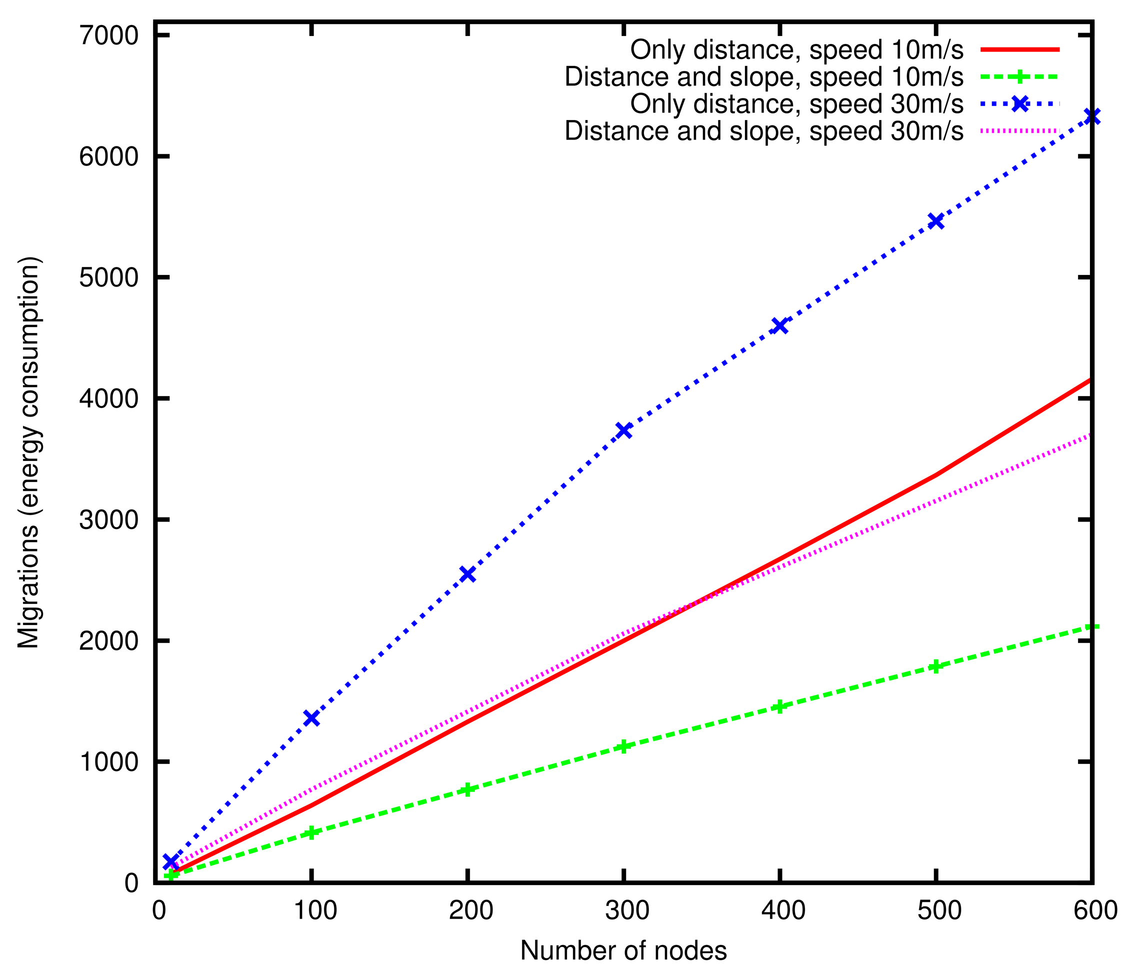

Figure 4: MA simulation results