Ad Hoc Networks 8 (2010) 462–473

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Using wireless sensor networks to support intelligent transportation systems ☆

David Tacconi [b,*], Daniele Miorandi [a], Iacopo Carreras [a], Francesco Chiti [b], Romano Fantacci [b]

[a] CREATE-NET, Via Alla Cascata 56/C, IT, 38100 Povo, Trento, Italy
[b] DET Department, Univ. of Florence, Via di Santa Marta, 50100 Florence, Italy

ARTICLE INFO

Article history:
Received 2 December 2008
Received in revised form 24 August 2009
Accepted 29 December 2009
Available online 7 January 2010

Keywords:
Disconnected wireless sensor networks
Vehicular communication
Mobility management
Energy aware protocols
Geographic routing
Performance evaluation

ABSTRACT

In this paper we propose a system architecture for enabling mobile nodes to query a largely deployed wireless sensor network in an intelligent transportation system scenario. We identify three different types of nodes in the network: mobile sinks (i.e. the nodes moving and querying the WSN), vice-sinks (i.e. nodes able to communicate directly with mobile sinks) and ordinary sensor nodes (i.e. nodes sensing a phenomenon and communicating in a multihop fashion). We present protocols and algorithms specifically tailored to such a scenario, in particular at the MAC and network layers. Such a reference architecture well covers situations in which WSNs deployed in a parking place or along a road, provide to cars information on the conditions of the surrounding environment. We introduce and analyse a simple geographic routing protocol and two different load balancing techniques. The performance of the proposed solutions is evaluated through extensive simulations. The simple geographic routing is compared to load balancing techniques. Results support the capability of the proposed solutions to enable the introduction of novel intelligent transportation system applications.

1. Introduction

The fast-growing field of Intelligent Transport Systems (ITS) spans from flight transport and traffic management to in-vehicle services like driver alert or traffic monitoring. As a consequence, transportation information collection and communication plays a key role in all intelligent transport application. Unfortunately, most conventional ITSs can only detect the vehicle in a fixed position, and the costs for deploying communication wires and power cables represent a barrier for their widespread adoption. In fact, nowadays, collecting traffic data for traffic planning and management is achieved mostly through wired sensors. The equipment and maintenance cost and time-consuming installations of these existing sensing systems prevent large-scale deployment of real-time traffic monitoring and control. Wireless sensor network (WSN) [3,4], thanks to the competitive advantages offered in terms of ease of deployment and maintenance offer the potential to significantly improve the efficiency of existing transportation systems, as described for instance in [5,6].

A WSN is typically constituted by: (i) a set of resource-constrained nodes, which are deployed over the spatial region to be observed and capable of performing user-defined tasks related to data gathering and (ii) one (or multiple) sink node(s), where information gathered by the WSN can be accessed by the end-user. Often WSNs are deployed in remote and/or hostile regions, and the sink is the only node through which the WSN is first queried

☆ Part of this work appeared in the Proc. of IEEE ICC 2007 [1] and in Proc. of IEEE Globecom 2007 [2]. The work of D. Miorandi and I. Carreras has been partially supported by the EC within the framework of the BIONETS project EU-IST-FET-SAC-FP6-027748, http://www.bionets.eu.

* Corresponding author. Tel.: +39 3478683464.
E-mail addresses: david.tacconi@gmail.com (D. Tacconi), daniele.miorandi@create-net.org (D. Miorandi), iacopo.carreras@create-net.org (I. Carreras), francesco.chiti@unifi.it (F. Chiti), romano.fantacci@unifi.it (R. Fantacci).

1570-8705/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.adhoc.2009.12.007

and then accessed for data gathering operations. Hence, the sink is expected to be connected to the backend through some form of long-range connection (i.e., satellite communication, Wi-Fi, Wi-Max, etc.).

In ITSs, this may result extremely inefficient in terms of infrastructure to be deployed, and management overhead in order to guarantee the correct operations of the network. In fact, when considering an urban setting, the scenario differs significantly from the previous one. Indeed, let us assume the WSN to be deployed along a road, or in city area covered by a specific pervasive service. Let us further assume the end-user, while moving around the city, to be in direct contact with the WSN, thus acting both as end-user and sink at the same time. In this case, the mobile user can directly access services offered by the ubiquitous WSN, without the need to resort to a remote repository where data is first stored, and then accessed. This is the case of ITSs, where deployed WSNs enable cars to obtain information on the conditions of the surrounding environment, and to use this information for taking appropriate decisions. The WSN could be for instance used for detecting the formation of ice over the road, or monitoring the status of parking spots along the streets of the city center.

This application domain, while intuitively clear, requires a general refinement of the standard WSN architecture and protocols. In particular, we are moving from a centralized architecture, where nodes self-organize at bootstrap phase for the delivery of data to the sink node, to a fully distributed one, where no pre-determined gateway can be identified.

In this work, we refine the system architecture initially proposed in [1,2], and specifically tailored to the support of WSNs in ITS operated in urban settings. The considered scenario consists of a WSN deployed over an urban area. One or multiple mobile sinks inject queries into the WSN, which answers, later on, with the requested information, if available. No particular requirement is imposed over the WSN deployment geometry, and packets are routed within the network according to a predictive geographical routing mechanism, where the position of final destination (sink) is adapted to the mobility pattern of the mobile user querying the network. This adaptation process is achieved through a mobility prediction scheme, which takes into account speed variations, sudden direction changes, etc., when forwarding packets.

Furthermore, we address here the network load balancing problems highlighted in [2]: we have shown that mobility management strategies and geographic forwarding stress a particular subset of the nodes in the network and expose them to energy failure. In order to prolong network lifetime, we introduce here an energy-aware forwarding strategy and a delay-aware forwarding strategy. In these strategies, nodes take decision on the next hop not only on the basis of geographic information, but also on the energy consumed by the neighbour or on the delay expected when sending a packet towards another node. In order to do that, the metric considered in simple geographic forwarding (i.e., a scalar product among the target direction and the neighbour direction) is weighted by the consumed energy in energy-aware forwarding and with the delay in delay-aware forwarding. In such a way we expect the more stressed nodes to be avoided from a certain point on and the performance to be improved.

In the remainder of this work we will present the considered reference scenario and the related system architecture in Section 2 and describe the routing framework in details in Section 3. We present then the load balancing techniques designed to work on top of the described routing framework in Section 4. The performance of the proposed system architecture, and related algorithms, is analysed by means of simulations in Section 5, focusing in particular on how the sinks mobility impacts the latency on the delivery of data and on the energy efficiency. Related works are presented in Section 6 and conclusion and future directions in Section 7.

2. System architecture

The operative scenario we refer to is constituted by an *information retrieval* area, where nodes sensing a phenomenon of interest are deployed, with one or more mobile users moving around it and querying for data. As an example, this includes the case of cars moving around at rush hours, and looking for a free parking spot in a particular area (i.e., close to a subway station, close to user's favorite shop). To this purpose a car acts as a sink: along its trajectory it gets temporarily connected to the network, it sends a query asking for information on certain geographical region, and then waits for the corresponding data.

In the aforementioned scenario we can identify three different roles, as shown in Fig. 1:

- *Sensor nodes (SNs)*, in charge of "sensing" a certain zone. We assume a large number of SNs to be deployed around a building in a block of a city town or in a trading center, detecting a parking lot around the building to be free or occupied. Furthermore, we assume each SN to be aware of its geographical position, for instance by storing in each node its coordinates during deployment. Alternatively, a distributed localization scheme can be used, where network nodes cooperate in order to reconstruct their spatial distribution [7].
- *Vice sinks (VSs)* that constitute the edge nodes managing the communications from and to MSs. We assume each VS to be aware of its position and of the position of the closest VSs and to have a unique progressive identifier (ID).

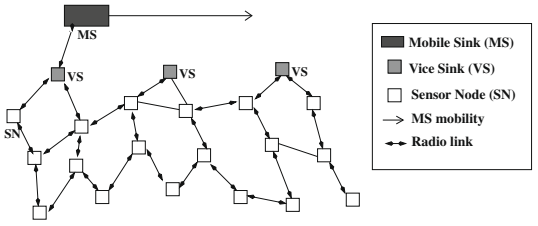

Fig. 1. System architecture: a mobile sink (MS) querying a WSN while moving. The nodes close to the streets, called vice-sinks (VSs), are in charge for communicating with MSs. The nodes in the inner WSN, called sensor nodes (SNs), communicate in a multihop fashion to reach VSs.

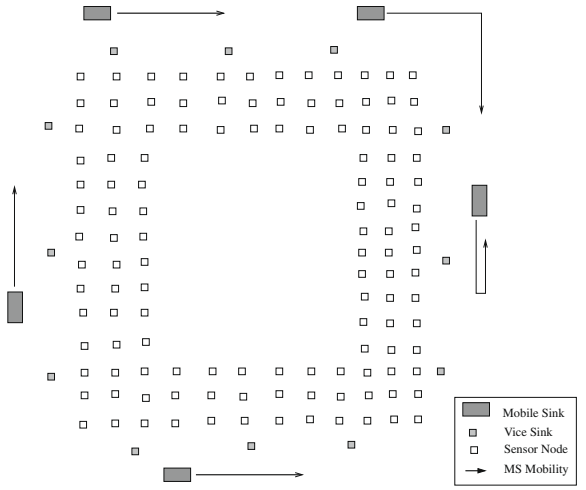

Fig. 2. An example scenario: cars move around a WSN deployed around a building and look for a free parking lot.

- *Mobile sinks (MSs)*, comprising the nodes moving along the deployment area where the *VSs* are deployed. We assume each MS to be equipped with a satellite receiver like GPS, so that information on position, direction and speed is always available.[1]

VSs are disseminated along the WSN network perimeter, but no particular constraint is applied to their density. In particular, it is not assumed that the MS is always reachable, thus resulting in several disconnections experienced from the MS, and it is not assumed that the VSs form a subnetwork, as they are not necessarily connected to each other. MSs are assumed to move according to a *constrained* random waypoint mobility model, where their position is constrained to stay along the peripheral area inside which the WSN is deployed. This includes the case of mobile users driving around a city block, and looking for a free parking spot as depicted in Fig. 2.

The communication architecture needs to be designed in such a way that a MS is allowed to send a query and to receive related responses, managing in a transparent way possible disconnections. It implies defining proper interfaces among MS, VSs and SNs.

3. Routing framework design

The routing framework we propose for the outlined architecture is based upon a geographic routing forwarding strategy enhanced with mobility prediction. In fact, after a query is injected in the network by a MS, a response message is expected to reach the outer nodes of the network by predicting the new position of MS, according to the mobility information included in the original query message. Typically, the VS node closest to the estimated position will be reached by the response packet. Then, if the MS is effectively in local proximity of the VS, the response is delivered with success, otherwise the packet needs to be routed towards the most likely actual position of the MS. In order to support that, we propose a geographic forwarding strategy coupled with an efficient mobility prediction scheme, able to use, at the VSs, the latest mobility information available at the MS.

The reference scenario is summarized in Fig. 3, where the MS injects a query in the WSN through the first VS. The query is then forwarded to the interested region (highlighted in grey) where the destination node (the closest to the center of the region) aggregates data of the region of interest by querying other nodes belonging to the same region. The aggregated data is then sent towards the target destination. The requested information is first forwarded from the SNs by means of multihop communications, and, finally, delivered from the last VS to the mobile sink.

3.1. Packets format

Before going into details of the routing strategy, let us introduce five different types of packets needed in order to support both the geographic forwarding and the mobility prediction strategies. The packets, their fields and the actors in the network managing them are the followings:

- *HELLO* packet: a simple packet sent periodically containing node's ID, geographical position, a flag to specify whether it is a VS node or not, the remaining energy (used for energy-aware forwarding) and the current duty cycle (used for delay-aware forwarding).
- *MOBILITY* packet: a simple packet sent by the MS to every VS in local proximity, containing direction of movement, geographic coordinates, speed and a global timestamp.
- *ALERT* packet: a message generated by every VS upon notification by a MS of a change occurred in its mobility pattern. It contains all the information contained also in the MOBILITY packet, plus the ID of the originating VS, the network address of the sender of the packet and the geographical coordinates of the destination of the ALERT packet.
- *QUERY* packet: a message generated by the MS after selecting the target region of the query itself. It contains the mobility information as in the MOBILITY packet, the geographical coordinates of the center of the target region and its radius of interest, the network address of the sender and the TTL of the packet.
- *REPLY* packet: a message generated by the SN closest to the center of the target region of the QUERY packet that contains all the mobility information of the originating MS as in the MOBILITY packet (copied from the QUERY packet), the actual position of the MS evaluated hop by hop according to mobility information and elapsed time, the network address of the sender and the TTL of the packet (copied from the QUERY packet).

The way these packets are handled by the routing framework is described in the following subsections.

[1] This is not considered limiting, given the widespread adoption of car navigation systems and the increasing inclusion of GPS receivers in mobile phones.

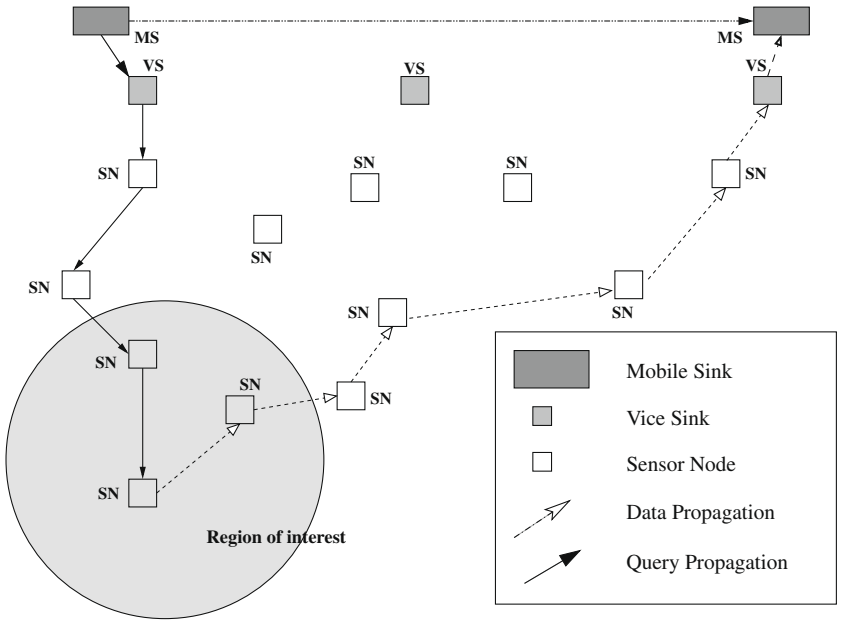

Fig. 3. Example of the considered application scenario. The mobile sink injects a query in the WSN through the closest vice sink. The query is forwarded to the queried region (highlighted in grey). The queried data is finally delivered through another VS.

3.2. Geographic forwarding

As stated in Section 2, we assume each node of the network to be aware of its position and each MS to be enabled with a satellite receiver such that it is able to know its coordinates, speed, direction and global timestamp. For the sake of simplicity, let us identify the coordinates of the target region stored in the QUERY packet with *TargetPos*, the coordinates of the MS stored in the REPLY packet with *MsPos* and the coordinates of the node that is currently taking the decision on the next hop with *CurrentPos*.

We describe the routing framework by separating the phases shown in the following.

3.2.1. Network topology construction

In the network bootstrap phase each SN builds its one-hop neighbour table by means of reciprocal HELLO packets exchange. Every node at bootstrap sends a *HELLO* packet described in Section 3.1. HELLO messages are scheduled at random instants in order to avoid collisions. Once the bootstrap phase is over, every node has created a routing table containing neighbourhood's geographical information. After the bootstrap phase is completed, each node periodically evaluates its remaining energy and its current duty cycle, stores this information in the next *HELLO* packet and then periodically broadcasts it. In such a way, each node in the network is aware of the position, the energy and the current duty cycle of its neighbours.

3.2.2. Packet forwarding strategy

A greedy forwarding strategy is applied, by selecting at each hop the neighbour that points closest to the direction of the intended destination. This is accomplished by letting each node forward the packet to the node i that maximizes the scalar product[2]:

$$\varphi_i = vers(neighbourPos_i, targetPos) \cdot vers(currentPos, targetPos) \quad (1)$$

where φ_i is the scalar product evaluated among the versors[3] of the current node position *currentPos* and the ith neighbour position $neighbourPos_i$ with the target position *targetPos*.

When a node finds out that the next hop coincides with the previous one, a DEADLOCK exception is thrown and the forwarding strategy enters the *back-up* mode. The current node stores in the packet the incoming direction in a field named *back-up angle* (i.e., the angle among the previous and the current node) and selects the neighbour that maximizes the scalar product with the versor in this direction.

At every step then, if the *back-up angle* is set, a node tries at first to find a neighbour that maximizes the scalar product towards the destination, otherwise if another DEADLOCK occurs, it keeps on following the *back-up angle* direction previously set, in order to overcome the hole. When a neighbour closer to the destination and different

[2] The scalar product (also referred to as 'dot product') is a standard operation in algebra; in its most current form it takes two vectors defined over a n-dimensional Euclidean space $\mathbb{R}^n$ (where $\mathbb{R}$ denotes the set of real numbers), and returns a real-valued scalar quantity, corresponding to the sum of the component-by-component multiplication between the two vectors. When applied to two versors, it returns the norm of the projection of the first versor over the second one. In such a case, it takes values in the interval $[0, 1]$, and equals 0 if the two versors are orthogonal, and 1 if they are parallel.

[3] A versor $vers(x, y)$, where x and y are points in a Cartesian space, is a vector of unitary norm having orientation equal to that of the vector pointing from x to y.

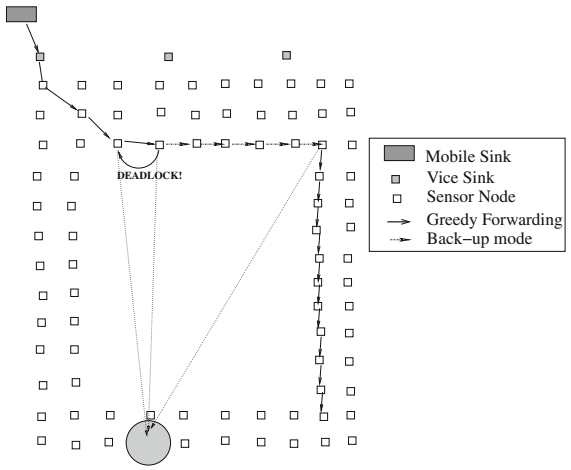

Fig. 4. A deadlock occurring when forwarding the packet towards a target region and founding a hole: the scalar product would be maximized by the previous hop, so the *back-up* mode is entered and the packet is forwarded around the hole. The *back-up* angle is reset when a node different from the previous hop that maximizes the scalar product towards the destination is found, which happens once the hole is overcome.

to previous hop is found (i.e. the hole is overcome) the *back-up angle* can be reset and nodes keep forwarding the packet towards the destination. This simple strategy allows each packet to reach the target region avoiding holes and deadlocks. The forwarding strategy and the DEADLOCK event are shown in Fig. 4.

3.2.3. Query propagation

A MS sends a query specifically to obtain information about a selected region. The region is specified by geographic coordinates and the query is forwarded toward the center of that region by each node according to the described packet forwarding strategy.

Using a SQL-like syntax, Algorithm 1 presents an example of a query injected by a MS in the WSN.

Algorithm 1 Example of a query injected in the wireless sensor network by the mobile sink

```
SELECT parkingInfo
FROM sensors
WHERE region C_r, R_r
MOBILITY POS_MS, V_MS, DIR_MS, T_MS
```

Once the query is accepted by the closest VS, the MS continues moving along the road, while expecting to receive the requested information in a reasonable amount of time. The *targetPos* (i.e. the center of the target region) of a QUERY packet selected by the MS remains unchanged after each hop of QUERY packet forwarding. Once the destination is reached by the query, the forwarding process is stopped and the source prepares to send the requested information. Mobility information regarding the original MS will be used to re-route the response toward the new position of the MS.

3.2.4. QUERY response

The *MsPos* for REPLY packet's forwarding is evaluated hop by hop according to the mobility information sent by the MS and originally included in the QUERY packet. The adaptive routing strategy implements the following operations at each SN node:

1. Evaluate the target destination based on the MS mobility information and the actual time.
2. Prepare the REPLY packet to forward including MS mobility information, data and next hop ID.
3. Check among the neighbours if there is a VS; if there is one, then send the message towards it, if no select the closer node to the target destination according to the packet forwarding strategy.

3.2.5. Information delivery

Only VSs are responsible for delivering information to MSs. Once the information has reached a VS, if the MS has not already passed by the current VS, a timestamp is set in order to wait for the MS for a reasonable time. If instead the MS has already passed by, the VS will use the received mobility information to re-route the packet towards the next target destination. The packet will reach the SN one hop further, and following the previously described strategy, it will go through the SNs in the direction where MS is moving, till the packet will be received by the next VS. This process will be iterated till an application dependent timestamp expires. This can happen for highly irregular movements of the MS.

The whole geographic forwarding strategy is summarized in Algorithm 2.

Algorithm 2 Packet forwarding strategy

```
receive msg;
if (msg is HELLO)
  update neighbours table;
else if (msg is QUERY)
  find next hop;
  forward QUERY to next hop;
else if (msg is REPLY)
  find next hop;
  if (TypeOfNode is VS)
    if (has updated mobility info)
      update mobility info;
      find next hop;
      forward REPLY to next hop;
    else
      store msg for a given time;
  else
    forward REPLY to next hop;
endif
```

3.3. Mobility management

The main goal of the mobility management mechanism is to the inform the VSs with the latest mobility information on the MSs. This is accomplished exploiting the fact

that REPLY packets will certainly reach the outer part of the WSN and then the VS closer to the estimated position of the MS. If the MS is not directly reachable by this VS, an appropriate decision on REPLY packet forwarding has to be taken. For this purpose, mobility information is sent by MSs every time they can communicate with a VS. In particular a MOBILITY packet is sent including fresh information about position, speed, direction and global timestamp. Each VS maintains this data in a structure that is updated upon receiving a fresher packet. When a REPLY packet reaches a VS two different decisions can be taken:

1. If no information fresher than the one currently stored in the packet is present at the VS, the packet waits a predefined amount of time for the MS to pass by or for fresher information to arrive (we will show later on how this can be achieved).
2. If fresher information is present at the VS, the mobility fields of the REPLY packet are updated and the packet is immediately forwarded towards the new destination.

Since a MS can invert the direction of mobility or simply make a turn it is crucial that close enough VSs get informed about the new mobility information if they cannot be directly reached by a new MOBILITY packet. Therefore we have introduced an algorithm that is able to inform a given number of VSs about the new mobility information, so that a REPLY packet can efficiently be forwarded toward the appropriate destination after reaching a VS. Whenever a VS detects a drastic change of direction in the mobility pattern it alerts close by VSs with the new mobility information.

In particular, two situations may occur:

- If the MS informs a VS of a just occurred inversion of direction, the VS sends an ALERT packet with the new mobility information towards the VSs in the previous direction of the MS. In such a way, a REPLY packet routed to a destination where the MS is expected to be found (according to the original mobility information) is immediately forwarded in the opposite direction, therefore increasing the probability of success.
- If the MS informs the VS of a just occurred change of direction while keeping the same direction around the WSN (for instance it has turn to another side of the parking lot area), the VS informs the other VSs of the previous side of the occurred change, e.g. the VSs previously encountered by the MS. This helps a REPLY packet being forwarded towards one side to immediately being routed according to the new information.

It is clear that such a technique introduces an additional communication overhead, but at the same time it allows the management of critical situations with a higher message delivery ratio and a lower latency. However, a Time to Live (TTL) field for the packets needs to be properly set in order to avoid useless information to be propagated in the network. In this case, a user looking for a parking lot in a specific geographic region could consider the information to expire after a given amount of time; it then forwards a new query until the reply arrives. In such a way we are able to evaluate the time needed by each user to receive the queried information (i.e., to find a free parking) in different conditions of mobility and network topology, as we will show in the following section.

The combination of geographic forwarding and mobility prediction strategy is reported in Algorithm 3.

Algorithm 3 Strategy applied at every VS for REPLY packet forwarding when receiving mobility information

```
receive msg;
if (msg is MOBILITY || msg is ALERT)
  if (MS is connected)
    send stored REPLY to MS;
  else
    find next hop;
    send stored REPLY to next hop;;
endif
```

4. Load balancing techniques

Energy consumption is one of the main issues in WSNs and especially in large-scale deployment as the ones we consider in this work. A WSN is composed by several nodes that are battery-supplied and which cooperate for distributing and delivering sensed information to querying nodes. Ideally, all the nodes should consume the same amount of energy and should die almost at the same time. It is obvious that depending on the peculiar network deployment and topology, as well as on traffic load, some nodes are more stressed than others and happen to die first with a high probability. When a node dies, all the network has to re-configure itself, which in turn implies a high consumption of resources. Energy-aware strategies aim at reaching network balancing with smart forwarding strategies or efficient MAC protocols, prolonging in such a way network lifetime, i.e. the time before which the first node in the network dies.

Given the previously described routing framework, we propose now two different techniques for load balancing in our architecture: energy-aware forwarding, a strategy that works at the network layer and delay-aware forwarding, a cross-layer technique that involves the MAC layer operations as well. In particular, when taking a decision on the next hop, each node evaluates a metric $dist_{x-i}$ by taking into account energy consumption or packet delay and decide for the neighbour that minimized the value of $dist_{x-i}$.

4.1. Energy-aware forwarding

We have shown in Section 3.2 that each node decides the next hop of a message by maximizing the progress towards destination. Each node broadcasts its position at network bootstrap and collects information about its neighbours through *HELLO* packets exchange. We recall that each node periodically broadcasts its position together with information about its battery consumption as described in Section 3.2. In order to keep the proposed strat-

egy as general as possible, we further assume that energy consumption directly depends on the number of transmitted and received packets, i.e., at node i:

$$E_i = E_{init} \cdot (1 - N_i \cdot \alpha_{pkt}) \quad (2)$$

where E_i is the energy available at node i, E_{init} is the available energy at bootstrap, N_i is the number of received and transmitter packets at node i and α_{pkt} is the percentage of energy consumed at each transmission. By periodically broadcasting this information, each node is then aware of the available energy of its neighbours with a good approximation, depending on the HELLO packet period.

We introduce now a different metric for packet forwarding decision. Let us denote by φ the scalar distance evaluated as described previously. The distance $dist_{x-i}$ among node x and its neighbour i is then computed as:

$$dist_{x-i} = \varphi_{x-i} \cdot \left(\frac{E_{init} - E_i}{E_{init}}\right) = \varphi_{x-i} \cdot N_i \cdot \alpha_{pkt} \quad (3)$$

Next-hop decisions are then taken according to this metric.

4.2. Delay-aware forwarding

We introduce now a cross-layer strategy based on an adaptive duty cycle at each sensor node, according to its energy consumption. We refer to the A-MAC protocol described in [8], where the adaptive duty cycle δ_c is defined at each node according to the following metric τ:

$$\tau = \frac{T_{elap}}{T_{conf}} - \frac{E_{elap}}{E_{init}} \quad (4)$$

where T_{elap} is the elapsed time since network bootstrap, E_{elap} is the energy consumed since network boot strap and T_{conf} is the pre-configured network lifetime.

Ideally, τ is equal to 0for any node in the system and the network is completely balanced; when τ takes a positive value it means that the node is consuming less than expected, while when τ takes negative values the node is excessively stressed. δ_c is adaptively varied according to τ. In particular, as respect to a starting value of $\overline{\delta_c}$:

- δ_c is doubled when $\tau < 0$.
- δ_c is halved when $\tau > 0$.
- δ_c is maintained when $\tau = 0$.

Based on the methods introduced in [1], it is possible to dimension the duty cycle in order to achieve an expected transmission delay. Sensor nodes are assumed to have a cycle time of $C_T = 1$ s and a duty cycle $D_C \in [1\%, 11\%]$. Considering a bit rate of 250 Kb/s, we have accordingly introduced an average delay equal to 130 ms, that corresponds to a duty cycle $D_C = 1.5\%$. This value has been derived by considering the saturation condition in which the bit rate is equal to 150 Kb/s[4] and the packet rate (in the active state) is $P_R = 521$ pkt/s given a packet length of 36 bytes. The maximum delay can be computed as the inverse of the time-averaged packet data rate, which in turn is given by the product of P_R times the length of an activity period (which equals $C_T \cdot D_C$), resulting in 130 ms for a duty cycle of 1.5%. As we increase the duty cycle, the expected transmission delay decreases, while it grows when the duty cycle is reduced. As described in Section 3.2, a node broadcasts to its neighbours $delay_i$, i.e., the delay a node x should expect when transmitting a packet to a node i. We can now define a new metric:

$$dist_{x-i} = \varphi_{x-i} \cdot delay_i \quad (5)$$

If next-hop decisions are taken according to this metric, we expect a higher network balancing and the average latency to be reduced with respect to geographic forwarding.

It is worth remarking that, while our approach, as in [1] is based on the use of A-MAC [8], it can be extended to other commonly used MAC protocols for WSNs such as S-MAC and B-MAC [9] by dynamically adapting the corresponding duty cycle.

5. Performance evaluation

In order to evaluate the proposed system performance we resort to extensive numerical simulations performed using Omnet++, an open-source simulation tool [10]. We assumed N_s^2 sensors to be uniformly deployed around a hole placed in the center of a square as shown in Fig. 2 in accordance to a regular grid disposition, with the distance between two consecutive sensors being 25 m. The size of the hole is 500 m × 500 m. A number of equally spaced vice-sinks (VSs) are also installed along each side of the road, and the distance among VSs is fixed to 125 m.

Mobile users are moving with a speed that is uniformly distributed between a minimum and maximum value and invert direction of movement (clockwise or counterclockwise) every 30 s with a certain probability. The current speed of the mobile users is updated every 5 s. MSs move along the outer road and can communicate only with VSs and have a communication range of 25 m. Every time a MS reaches a corner of the outer road, it changes its direction, while preserving the movement pattern, either clockwise or counterclockwise.

A MS randomly selects the starting position in the outer square and the initial direction of movement. It also selects a region in the network and sends a *QUERY* packet toward it through a selected VS. Every time a *QUERY* is injected in the network the MS starts a timer and accordingly sets the Time To Live (TTL) field of the packet, after which it is dropped. The MS sends the same *QUERY* packet again until it does not receive *REPLY* before the timer expires. We have set the expiration time to 90 s. Nodes communicate at a rate of 250 Kb/s, and adopt CSMA/CA mechanism for managing medium access.[5] Also, the duty cycle operation has

[4] In fact, it is well known that a CSMA/CA medium access control mechanism such as the one encompassed in the IEEE 802.15.4 standard introduces an overhead of ≈40%.

[5] As remarked in the previous section, the framework can be extended to account for other MAC protocols widely employed in WSN deployments, such as S-MAC and B-MAC. The numerical results obtained would vary, but the comparison of the various routing strategies employed would still provide similar outcomes.

Table 1
Different network topologies with variable number of nodes N : $\#SN$ is total number of sensor nodes, $\#VS$ the number of vice-sinks, $\#VS_{alerted}$ the number of VS alerted by the mobility algorithm and $zone_x$ and $zone_y$ the dimensions of the network (in m).

N	$\#SN$	$\#VS$	$\#VS_{alerted}$	$zone_x$ (m)	$zone_y$ (m)
192	176	4	2	650	650
404	384	5	2	750	750
648	624	6	3	850	850
924	896	7	3	950	950
1232	1200	8	4	1050	1050

been taken into account by introducing an additional delay to each packet transmission.

We aim now at evaluating and comparing the performance of the proposed routing strategies while varying the size of the network and preserving MS mobility characteristics. For the sake of conciseness,from now on, the simple geographic routing is labeled as *Geo-forwarding*, the energy aware routing with *Energy-forwarding* and the delay aware routing with *Delay-forwarding*.

In addition to the proposed schemes, we have also evaluated the case of a basic flooding strategy, which differs from the previous schemes in the way the response is returned to the *MS*. In this case, the response is simply forwarded by each *SN* to its neighbours until a *VS* is encountered. When a *VS* is reached, the response message is stored until the final delivery to the *VS* (or deleted after a timeout).

The parameters under evaluation are:

- *Latency:* the time elapsed between transmission of a query and reception of a reply by a MS.
- *Hop-count:* the average number of hops traversed by each packet.
- *Time to first failure:* the time before which the first node in the network dies due to battery exhaustion.

We fixed the initial energy of a node to an adimensional value $E_{init} = 50$ and the percentage of consumed energy by a transmitted or received packet to $\alpha = 1\%$. Furthermore, the initial duty cycle is fixed to 1.5% with a corresponding delay of 130 ms and the network lifetime is estimated to be 5000 s, a parameter used in the delay-aware forwarding strategy. The MS moves at a minimum speed v_{min} of 5 m/s and at a maximum speed v_{max} of 20 m/s, while the probability of inversion $p_{inversion}$ is fixed to 0.5. The size of the network is increased by adding new lines of sensors around the central hole. Furthermore, we keep the distance among consecutive VS fixed, and consequently the number of VSs increases accordingly together with number of $VS_{alerted}$. The various network topologies with the corresponding simulation parameters are described in Table 1.

Fig. 5 shows the time to first failure with the four forwarding strategies. The results of the simulation have been obtained by running 50 different simulations for each value of N (i.e. the number of nodes in the network) and registering the time in which the first node consumed all its initial energy. It can be observed that, as the size of the network increased, the simple geographic routing overcomes load balancing techniques. In particular, for the biggest network under evaluation, *delay-forwarding*

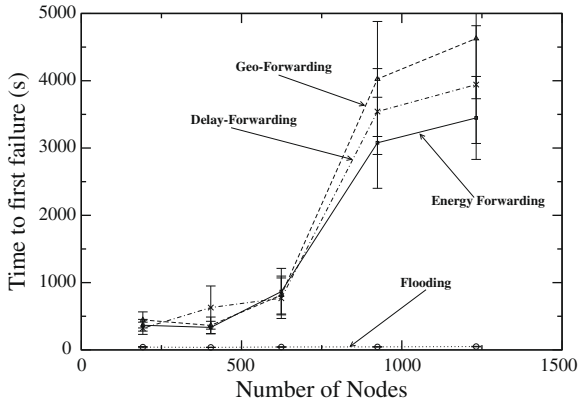

Fig. 5. Average time to first failure versus number of nodes N with confidence interval of 98%. v_{min} = 5 m/s, v_{max} = 20 m/s, $p_{inversion}$ = 0.5, VSs spaced by 125 m, 50 runs for each point.

has an average time to first failure 10% lower than that of *geo-forwarding*, while *energy-forwarding* is 20% lower. On the other side, for the smallest network size figures, the three strategies performs almost identically. In fact, in small network (i.e. $N \leqslant 648$) there are not enough possible routes toward the destination to find an alternative when a node is consuming all its energy, while in biggest network (i.e. $N \geqslant 924$) load balancing techniques can benefit from several paths and decide for less stressed nodes. Flooding is the scheme that performs worst in all situations. As expected, the large number of messages introduced for sending back the response to the *MS* are draining the *SNs* energy very quickly. This results in a much shorter network lifetime.

Figs. 6 and 7 depict the total number of packets and the corresponding overhead introduced by the Flooding scheme and the *geo-forwarding* one. This is evaluated by counting the number of total network packets associated to the delivery of a certain message. In line with the time to first failure results (Fig. 5), it is possible to observe that the Flooding scheme is generating a much higher total number of packets. Vice versa, geo-forwarding presents a much higher overhead, due to the control packets exchanged in order to set up optimized routes to the MSs. At the same time, such an overhead is well worth, as the total number of packets results lower than in the flooding case.

In Fig. 8 the packet latency is plotted for the same set of simulations. It can be observed that in this case, while *geo-forwarding* and *energy-forwarding* performs similarly,

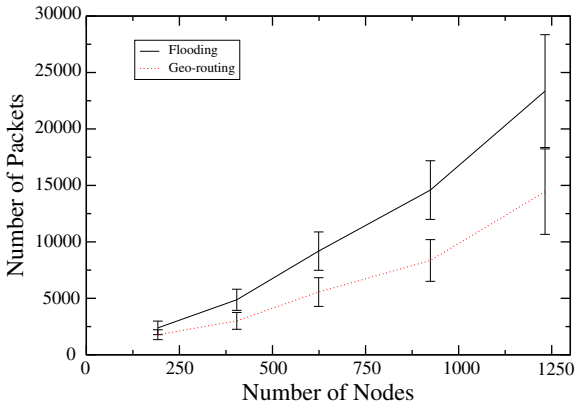

Fig. 6. Total number of packets versus number of nodes N with confidence interval of 98%. v_{min} = 5 m/s, v_{max} = 20 m/s, $p_{inversion}$=0.5, VSs spaced by 125 m, 50 runs for each point.

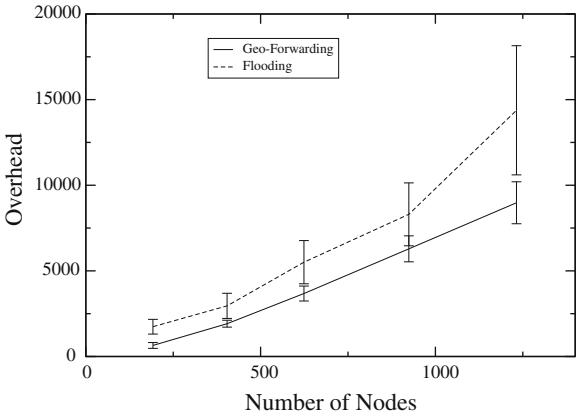

Fig. 7. Overhead versus number of nodes N with confidence interval of 98%. v_{min} = 5 m/s, v_{max} = 20 m/s, $p_{inversion}$ = 0.5, VSs spaced by 125 m, 50 runs for each point.

delay-forwarding decreases considerably the packet latency for any size of network. In fact, in this strategy the duty cycle is adaptively reduced or augmented according to consumed energy, and consequently, at the beginning of the simulation, less stressed nodes can forward packets with a reduced delay.

Fig. 9 shows the number of hops a packet has passed through between query injection and reply reception. It can be easily seen that the hop-count is approximately the same for the three strategies and that it increases with the size of the network. This implicitly shows that the routes used by *QUERY* and *REPLY* packets do not depend (at least in terms of length) upon energy consumption level across nodes. Furthermore, we can conclude that *ALERT* packets and in general mobility prediction affects the time to first failure parameter and highlight the difference among load balancing strategies and simple forwarding.

We are now interested in evaluating the performance of the three forwarding strategies when varying the mobility conditions of the MS. In particular, we keep fixed the size

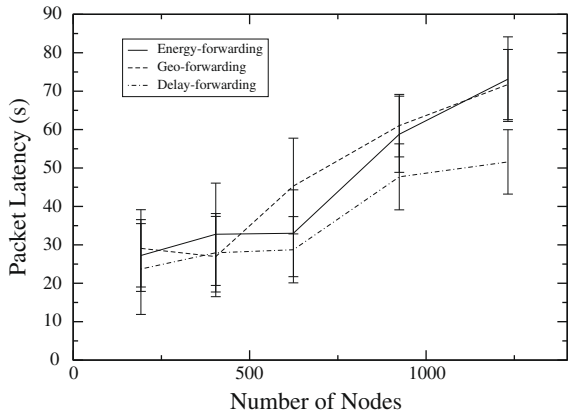

Fig. 8. Average latency of packets versus number of nodes N with confidence interval of 98%. v_{min} = 5 m/s, v_{max} = 20 m/s, $p_{inversion}$ = 0.5, VSs spaced by 125 m, 50 runs for each point.

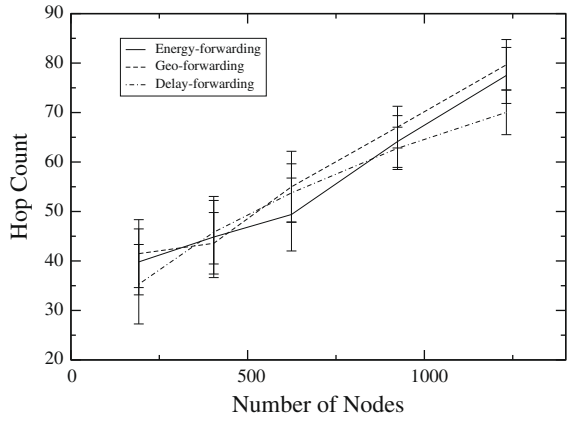

Fig. 9. Average hop-count versus number of nodes N with confidence interval of 98%. v_{min} = 5 m/s, v_{max} = 20 m/s, $p_{inversion}$ = 0.5, VSs spaced by 125 m, 50 runs for each point.

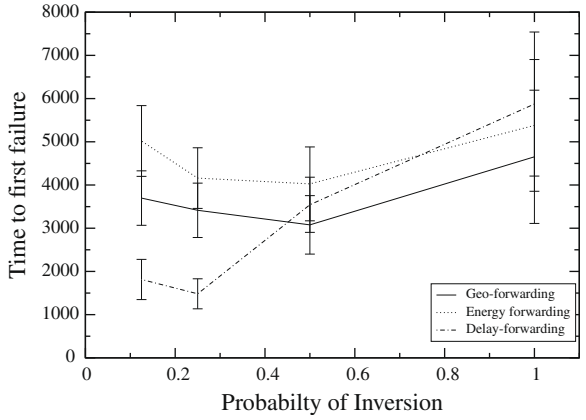

Fig. 10. Average time to first failure versus $p_{inversion}$ with confidence interval of 98%. v_{min} = 5 m/s, v_{max} = 20 m/s, N = 924, VSs spaced by 125 m, 50 runs for each point.

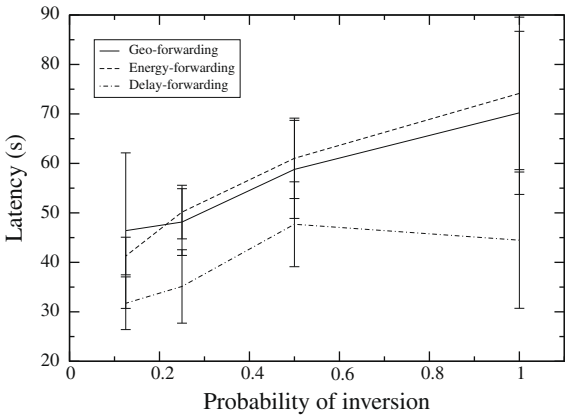

Fig. 11. Average latency versus $p_{inversion}$ with confidence interval of 98%. v_{min} = 5 m/s, v_{max} = 20 m/s, N = 924, VSs spaced by 125 m, 50 runs for each point.

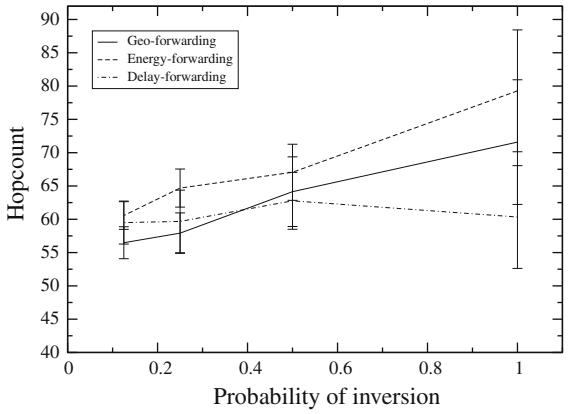

Fig. 12. Average hop-count versus $p_{inversion}$ with confidence interval of 98%. v_{min} = 5 m/s, v_{max} = 20 m/s, N = 924, VSs spaced by 125 m, 50 runs for each point.

of the network to $N = 924$, set V_{min} to 5 m/s, V_{max} to 20 m/s and we vary the probability of inversion from 0.125 to 1. In this set of simulation, we still measure the time to first failure, packet latency and packet hop-count.

Fig. 10 shows the *time to first failure* while increasing the probability of inversion at the MS. At first, we observe that *energy-forwarding* always outperforms *geo-forwarding*, while *delay-forwarding* has a shorter time to first failure for small network and a longer one for larger networks.

Then, it is interesting to observe that we have a time to first failure that decreases toward a floor value that for simple forwarding and energy-aware forwarding is reached at $p_{inversion} = 0.5$ while for the delay aware strategy is reached at $p_{inversion} = 0.25$. From this point on, we observe an increase in the time to first failure. This can be explained considering that, since for high $p_{inversion}$ values the node is often changing its direction while remaining close to a VS (we can think of it as performing a random walk around a given VS). Therefore, after notifying a VS of its position the MS does not perform an inversion before reaching another VS. In such a way, the alerting strategy does not starts and less packets are injected into the network, with a consequent energy saving.

At the same time, in Fig. 11 we observe how as increase in $p_{inversion}$ leads to an increase in packet latency, as well as in the hop-count, as it can be seen from Fig. 12. This is a confirmation of the above explained fact: by increasing the $p_{inversion}$, the mobility strategy injects into the network a smaller number of *ALERT* packets and in such a way, VSs that are reached by a reply packet happen to be not aware of the movement pattern of the MS. The *REPLY* packet starts then to follow the MS without information on its mobility and in such a way the latency of a packet is increased. This confirms the goodness of the mobility prediction strategy, which comes at the cost of a higher energy consumption.

6. Related work

There are several works addressing the problem of handling a mobile sink in wireless sensor networks deployed in the environment. We recall that differently from several other solutions and studies, our idea is to have a mobile user querying directly the WSN while it is moving, the network itself being in charge of routing the appropriate information back to the sink.

In SENMA [11] and MSSN [12] the mobile user gathers information through one-hop communications from either a sparse or dense WSN. SENMA (SEensor network with Mobile Agent) exploits nodes redundancy using mobile agents (e.g. mobile sinks) that collect data from WSN by means of single-hop communications. The application depicted for SENMA is an aircraft flying above a dense deployed sensor network that wants to collect data regarding the sensed environment or performs some network monitoring techniques. MSSN (mobile sink in sensor network) is based on the same concept of single-hop communications between sensor nodes of a sparse WSN and the mobile sink. The envisioned application is a police car moving along a highway to gather data from sensors deployed to collect traffic and surveillance information.

These work avoid the problem of multihopping and consequently of querying information and routing data by defining application where single-hop relaying is the only way for the sink to gather data from sensors; it is also stated that multihop communication have a negative impact in terms of energy consumption, although it regards the specific application scenarios depicted.

In [13–15] the sink acts as a mobile relay (i.e. data mule) that gathers data from a location and brings them somewhere else. In [16–18] the mobile node replaces a static node to let it save some energy. Both approaches have as final goal the maximization of the network lifetime exploiting sink mobility. In [19] an analysis of controlled mobility versus uncontrolled mobility is performed.

In Hyper [20] a routing strategy is designed to allow mobile users to gather information from the WSN while performing maintenance work or environmental evaluation. The goal is to relay packets to the mobile sink by efficient routing strategies, aiming at maximizing throughput and energy efficiency.

In TTDD [21] the issue of multiple mobile sinks in a large and dense WSN is considered. Differently from other works, TTDD divides the network in a grid where each node is location-aware and tries to minimize the overhead due to multiple and frequent queries from the sinks using a hierarchical approach. This protocol is proven to perform as efficient as commonly used routing protocols in the presence of fixed sinks, in terms of energy consumption, throughput and end-to-end delay.

When considering the area of ITS, WSNs have been proposed as a solution for gathering real-time information about road conditions [22]. In [23], the authors present a WSN for proactively detecting and advertising possible dangerous situations on roads. In [24], a traditional WSN architecture is optimized in order to achieve a high cars detection accuracy, while preserving a sufficiently high network lifetime. Several works in the field of ITS focus on enhanced parking management strategies by enabling each parking space with a sensor node. In [25] a combination of magnetic and ultrasonic sensors for accurate and reliable detection of vehicles in a parking lot is used. In [26] the status of the parking field detected by sensor nodes is reported periodically to a database via the deployed wireless sensor network and its gateway. Reported information can be then used by a centralized system to perform various management functions, such as finding vacant parking lots, automatic tolling, security management, and statistic report.

Finally, a market example is provided by Streetline technologies [27], a company located in San Francisco that deploys thousands of sensor nodes on parking lots and provides information on each parking lot (available, occupied or violated) to customers paying for the service. In this case the information is provided by a centralized system that collects all the data and send information to users' devices. We are here interested in investigating the case where customers are in direct contact with the deployed WSN, without the need to pass through a centralized server.

7. Conclusions

In this paper we have proposed a system architecture and a set of routing protocols for enabling ITS applications in which a mobile node queries data from a static WSN. This application scenario well fits the case of cars moving around a hot parking place area, where sensors able to detect a parking place to be free or occupied organize themselves in a WSN. MSs can in general experience periodical disconnections from the WSN, depending on the deployment of nodes along the road where they are moving. In order to efficiently react to unpredictable mobility pattern of the MSs we have described a geographic routing strategy able to overcome holes by means of a mobility prediction strategy.

We have introduced a system architecture based on three types of nodes: mobile sinks (MSs) i.e., mobile nodes querying the WSN, vice-sinks (VSs), i.e., nodes in charge of communicating with a MS, and sensor nodes (SNs), i.e., simple nodes that perform query and response forwarding but can reach MS only through a VS. We have introduced two simple load balancing techniques,which account for energy and delay, respectively, when taking routing decisions. Such strategies have been tested and compared to the simple geographic forwarding protocol for various settings (network size and MS mobility pattern). The proposed solutions have been tested against a simple flooding algorithm in order to evaluate their efficiency and the overhead due to the transmission of control packets.

Possible research directions include the design and testing of appropriate buffer management strategies. Data aggregation and data fusion algorithms shall also be in order to complete the study on the depicted application scenario. We are also aiming to develop a small test bed to evaluate the results with real hardware.

References

[1] D. Tacconi, I. Carreras, D. Miorandi, I. Chlamtac, F. Chiti, R. Fantacci, Supporting the sink mobility: a case study for wireless sensor networks, in: Proc. of IEEE ICC, Glasgow, Scotland, 2007.
[2] D. Tacconi, I. Carreras, D. Miorandi, F. Chiti, A. Casile, R. Fantacci, A system architecture supporting mobile applications in disconnected sensor networks, in: Proc. of GLOBECOM, Washington, USA, 2007.
[3] I. Akyildiz, I. Kasimoglu, Wireless sensor and actor networks: research challenges, Ad Hoc Networks 2 (4) (2004) 351–367.
[4] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, D. Culler, An analysis of a large scale habitat monitoring application, in: Proc. of SenSys, New York, NY, USA, 2004, pp. 214–226.
[5] J. Bohli, A. Hessler, O. Ugus, D. Westhoff, A secure and resilient WSN roadside architecture for intelligent transport systems, in: Proc. of WiSec, 2008, pp. 161–171.
[6] W. Chen, L. Chen, Z. Chen, S. Tu, Wits: a wireless sensor network for intelligent transportation system, in: Proc. of IMSCCS, Hangzhou, China, 2006.
[7] N. Patwari, J. Ash, S. Kyperountas, A.O. Hero, R.L. Moses, N.S. Correal, Locating the nodes: cooperative localization in wireless sensor networks, Signal Processing Magazine 22 (4) (2005) 54–69.
[8] Y. Nam, H. Lee, H. Jung, T. Kwon, Y. Choi, An adaptive MAC (A-MAC) protocol guaranteeing network lifetime for wireless sensor networks, in: Proc. of EW, Athens, Greece, 2006.
[9] J. Polastre, J. Hill, D. Culler, Versatile low power media access for wireless sensor networks, in: Proc. of ACM SenSys, New York, NY, USA, 2004.
[10] OMNeT++ Discrete Event Simulation System. <http://www.omnetpp.org>.
[11] L. Tong, Q. Zhao, S. Adireddy, Sensor networks with mobile agents, in: Proc. of IEEE MILCOM, Boston, MA, USA, 2003.
[12] L. Song, D. Hatzinakos, A cross-layer architecture of wireless sensor networks for target tracking, IEEE/ACM Transactions on Networking 15 (1) (2007) 145–158.
[13] R. Shah, S. Roy, S. Jain, W. Brunette, Data mules: modeling a three-tier architecture for sparse sensor networks, in: Proc. of IEEE SNPA, 2003.
[14] K. Lan, Z. Wu, On the feasibility of using public transport as data mules for traffic monitoring, in: Intelligent Vehicles Symposium, 2008 IEEE, 2008, pp. 979–984.
[15] G. Anastasi, M. Conti, M. Di Francesco, Data collection in sensor networks with data mules: an integrated simulation analysis, in: Proc. of IEEE ISCC, Marrakech, Morocco, 2008, pp. 1096–1102.
[16] W. Wang, V. Srinivasan, K. Chua, Using mobile relays to prolong the lifetime of wireless sensor networks, in: Proc. of ACM MobiCom, Cologne, Germany, 2005.
[17] W. Wang, V. Srinivasan, K. Chua, Extending the lifetime of wireless sensor networks through mobile relays, Networking, IEEE/ACM Transactions on Networking 16 (5) (2008) 1108–1120.
[18] J. Luo, J. Panchard, M. Piorkowski, M. Grossglauser, J. Hubaux, Mobiroute: routing towards a mobile sink for improving lifetime in sensor networks, Lecture Notes in Computer Science 4026 (2006) 480.
[19] S. Basagni, A. Carosi, C. Petrioli, Controlled vs. uncontrolled mobility in wireless sensor networks: some performance insights, in: Proc. of IEEE VTC-2007, 2007, pp. 269–273.

[20] D.E. Thomas Schoellhammer, Ben Greenstein, Hyper: A Routing Protocol to Support Mobile Users of Sensor Networks, Tech. Rep., CENS, January 2006.
[21] H. Luo, F. Ye, J. Cheng, S. Lu, L. Zhang, TTDD: two-tier data dissemination in large-scale wireless sensor networks, Wireless Networks 11 (2005) 161–175.
[22] F. Zhao, L. Guibas, Wireless Sensor Networks: An Information Processing Approach, Elsevier/Morgan-Kaufmann, 2004.
[23] M. Karpinski, A. Senart, V. Cahill, Sensor networks for smart roads, in: Proc. of IEEE PerCom Workshops, Pisa, Italy, 2006.
[24] S. Coleri, S.Y. Cheung, P. Varaiya, Sensor networks for monitoring traffic, in: Proc. of Allerton Conf., 2004.
[25] S. Lee, D.Y.A. Ghosh, Intelligent parking lot application using wireless sensor networks, in: Proc. of CTS, Irvine, CA, USA, 2008.
[26] V. Tang, Z. Yuan, C. Jiannong, An intelligent car park management system based on wireless sensor networks, in: Proc. of ISWPC, Phuket, Thailand, 2006.
[27] Streetline, City Infrastructure Technologies, <http://www.streetlinenetworks.com/>.

David Tacconi was born in Florence in 1979. He received the telecommunications engineering degree from the University of Florence in 2003. He did his diploma thesis at New Jersey Institute of Technology on routing protocols for 4 G network. After completion of his degree his was hired by Telecom Italia and focused on network monitoring systems and tools. Then, he has been with Create-Net research center working in the area of Pervasive computing while pursuing his PhD at the University of Florence. Currently he is a Project Manager in the area of application and services at Futur3 s.r.l. a start-up wireless telecommunication company. His general research interest includes wireless sensor networks, pervasive computing, wireless mesh networks, and mobile services and communications.

Daniele Miorandi was born in Rovereto, Italy, in 1977. He received the laurea degree (summa cum laude) in Telecommunication Engineering from the University of Padova, Italy, in 2001, with a thesis on spectral synthesis using finite state machines. He received the PhD degree in Communication Engineering from the University of Padova, Italy, in 2005, with a thesis entitled "Stochastic Modelling of Wireless Ad Hoc Networks". In 2003/04 he spent one year of his doctoral thesis visiting the MAESTRO project at INRIA Sophia Antipolis (France). In 2004 he had an appointment as "Incaricato di Ricerca" at IEIIT-CNR, Torino (Italy). In January 2005 he joined the Pervasive team at CREATE-NET, Trento (Italy). Since October 2006 is the Head of the Pervasive area. Since March 2007 he is the coordinator of the EU-funded BIONETS project.

Iacopo Carreras received the telecommunications engineering degree from the University of Pisa in 2001. He did his diploma thesis at Technische University of Berlin in the topic VoIP over 802.11 wireless LANs. After completion of his degree his was hired by Netikos and focused on IT service provisioning for TLC operators. Currently, he is at create-net research center working in the area of Pervasive computing and pursues his PhD at University of Pisa. His general research interest includes wireless technologies, pervasive computing, bio-inspired computing, and mobile communications.

Francesco Chiti (M'01) received the degree in Telecommunications Engineering and the PhD degrees in Informatics and Telecommunications Engineering from the University of Florence in 2000 and 2004. His current research topics are devoted to Link and Network layers protocols design for Ad Hoc and sensor networks. He took part in several European research projects as IP GoodFood, STREP DustBot, NoEs NEWCOM and CRUISE, GJU TWIST, ETSI STF179 and COST 289 action.

Romano Fantacci, (M'87,SM'91, F'05) born in Pistoia, Italy, graduated from the Engineering School of the Universitá di Firenze, Florence, Italy, with a degree in electronics in 1982. He received his PhD degree in telecommunications in 1987. After joining the Dipartimento di Elettronica e Telecomunicazioni as an assistant professor, he was appointed associate professor in 1991 and full professor in 1999. His current research interests are digital communications, computer communications, queuing theory, satellite communication systems, wireless broadband communication networks, adhoc and sensor networks. He has been involved in several European Space Agency (ESA) and INTELSAT advanced research projects. He is the author of numerous articles published in prestigious communication science journals. He guest edited special issues in IEEE Journals and magazines and served as symposium chair of several IEEE conferences, including VTC, ICC and Globecom. Professor Fantacci received the IEE IERE Benefactor premium in 1990 and IEEE COMSOC Award Distinguished Contributions to Satellite Communications in 2002. He is currently serving as Editor for Telecommunication Systems, IEEE Trans. Commun. and IEEE Transactions on Wireless Communications.