Markovian Agents modeling swarm intelligence algorithms in wireless sensor networks

Dario Bruneo[1], Marco Scarpa[1], Andrea Bobbio[2], Davide Cerotti[2], Marco Gribaudo[3]

[1]*Dipartimento di Matematica, Università di Messina, Messina, Italy;*
[2]*Dipartimento di Informatica, Università del Piemonte Orientale, Alessandria, Italy;*
[3]*Dipartimento di Elettronica ed Informazione, Politecnico di Milano, Milano, Italy*

Abstract

Wireless Sensor Networks (WSN) are large networks of tiny sensor nodes that are usually randomly distributed over a geographical region. The network topology may vary in time in an unpredictable manner due to many different causes. For example, in order to reduce power consumption, battery operated sensors undergo cycles of *sleeping - active* periods; additionally, sensors may be located in hostile environments increasing their likelihood of failure; furthermore, data might also be collected from a range of sources at different times. For this reason multi-hop routing algorithms carrying messages from a sensor node to a sink should be rapidly adaptable to the changing topology. Swarm intelligence has been proposed for this purpose, since it allows the emergence of a single global behavior from the interaction of many simple local agents. Swarm intelligent routing has been traditionally studied by resorting to simulation. The present paper aims to show that the recently proposed modeling technique, known as *Markovian Agents*, is suited for implementing swarm intelligent algorithms for large networks of interacting sensors. Various experimental results and quantitative performance indices are evaluated to support this claim. The validity of this approach is given further proof by comparing the results with those obtained using a WSN discrete event simulator.

Keywords: Wireless Sensor Networks, Markovian Agents, Swarm intelligence, Gradient-based routing, Performance evaluation.

1. Introduction

Wireless Sensor Networks (WSN) are application-specific networks composed of a multitude of tiny sensor nodes with limited computational, communication, and power capabilities [2]. Sensor nodes collect measurements of physical parameters and transmit them to a sink node. Sensors may be scattered randomly over a geographical region and, in order to save battery energy, they may undergo

Email address: [1]`[dbruneo,mscarpa]@unime.it`, [2]`[bobbio,davide.cerotti]@mfn.unipmn.it`, [3]`gribaudo@elet.polimi.it` (Dario Bruneo[1], Marco Scarpa[1], Andrea Bobbio[2], Davide Cerotti[2], Marco Gribaudo[3])

A preliminary version of this paper has appeared as [1]

cycles of *sleeping - active* periods [3]. Nodes deployed in real fields might get damaged, or just fail at any time. The sink node might also change its location, and more than one sink can be present at the same time. As a result, the topology of the active nodes in a WSN may vary in time in an unpredictable manner. For this reason routing algorithms used to carry messages from a sensor node to a sink in a multi-hop fashion should rapidly adapt to the changing topology. A survey of routing algorithms is in [4].

Swarm intelligence (SI) techniques [5] are population-based stochastic methods in which the collective behavior of relatively simple individuals arises from their local interactions to produce global patterns. Through the adoption of the swarm intelligence concept, it is possible to design distributed, self-organizing, and fault tolerant routing protocols able to self-adapt to environmental changes. The main properties of SI-based routing protocols are that: *i)* single nodes are assumed to be simple with low computational and communication capabilities; *ii)* nodes are not aware of their position and communicate indirectly, i.e., messages are not directed to any particular node; *iii)* the range of messages may be very short, nevertheless a robust global behavior emerges from the interaction of the nodes; *iv)* global behavior adapts to topological and environmental changes.

SI in WSN is inspired by observing on how ant colonies forage for food [5]. Ants release a substance called *pheromone* during their passage, and tend to move along paths where a high pheromone trace is present, reinforcing that specific route. However, pheromone evaporates allowing the system to remove existing information and randomly search for new solutions. In this way, large groups of simple agents, interacting only locally with neighboring agents, work together to coordinate their actions toward fulfilling a common goal. In such systems, modeling the state space of the entire system as a cross-product of the state spaces of individual nodes results in the well-known state explosion problem. In fact, the usual way to study these systems is through simulation [6, 7]. A few papers proposes analytical approaches, as surveyed in Section 8.

This paper describes how the performance analysis of large SI-WSN systems composed of interacting agents can be modeled and analytically evaluated by resorting to a recently defined new entity called *Markovian Agents (MA)* [8, 9]. A MA is a discrete-state continuous-time Markov chain (CTMC) governed by a transition rate matrix that contains local rates and interaction-induced rates. MAs interact by sending and receiving messages that modify their behavior. Furthermore, agents are located in a geographical space, and their interaction depends on their relative positions and is governed by a suitable *perception function*.

In [10] the MA formalism was extended to include the capability of exchanging several types of messages. The present paper illustrates a further extension to this development, introducing a set of classes, each of them describing a different agent behavior. SI-WSN systems are thus modeled by two types of MAs: one for the sinks and one for the sensors. The paper describes a stochastic model to analyze a swarm-based routing protocol that is inspired by the one presented in [6]. According to [6], pheromone information is stored at each node, and the algorithm starts with the sink agent(s) emitting a message with the highest pheromone level; sensor agents that receive the pheromone message update their pheromone level and transmit it to their neighbors; at the same time sensor nodes are subject to an evaporation process that reduces their stored pheromone intensity. The assumption is made in this paper that the nodes (either sinks or sensors) are scattered over a rectangular mesh, with at most one node in each cell. Even if the transmission range of each node limits the activity of the pheromone messages to the closest neighboring nodes, the pheromone gradient rapidly forms over the entire region. Consequently, the main aim of this paper is to show that a large system of interacting MAs can be analytically solved to generate a pheromone intensity distribution around the sink(s) that can be successively used

to create the routing table along the steepest gradient in order to minimize the number of hops from each node to the sink(s).

A formal analytical model for the SI-WSN system is presented and the related analytical solution is illustrated. It can be proven that the implemented numerical technique is very efficient and scalable and can cope with scenarios composed of thousands of sensor-nodes and tens of sinks in conditions in which a state-space based analysis is unusable. It is noted again that a stable pheromone gradient is established by letting each MA interact only with its first neighbors. This limits as much as possible the transmission range and reduces the number of exchanged messages as well as energy consumption. Several examples illustrate the sensitivity of the solution to characteristic parameters, such as emission and evaporation rates, transmission range, and different topological configurations. By increasing the transmission range, the system reaches a stable pheromone configuration faster but at the cost of increased energy consumption. Finally, the analytical model is validated via simulation.

The paper is organized as follows. Section 2 briefly revises routing protocols for WSN and describes the swarm-based routing algorithm. Section 3 presents the Markovian Agent technique and illustrates its analytical and numerical solution. The SI algorithm is modeled in Section 4. Section 5 computes some performance indices that characterize the system. Section 6 is devoted to validating the model and Section 7 provides an extensive set of experiments. Finally, an overview of the related works is presented in Section 8, while Section 9 presents the overall conclusions.

2. Gradient-based routing protocols in WSN

WSN's specific characteristics make routing different from traditional wireless ad-hoc networks [2, 11, 4]. Various routing metrics can be exploited with respect to different goals: hop count, energy consumption, Quality of Service, throughput, network lifetime [12, 13, 14]. However, the main philosophy is that the information the nodes contains is more important than the nodes themselves. Such assumption leads routing in WSN to become data-centric as opposed to node-centric [11]. In this context, gradient-based routings [15, 12] allow to establish routes to the sinks following such paths that respect criteria related to data typologies, network topology, and nodes' status.

From an autonomic perspective, routing algorithms easy to manage and able to react to the environmental changes are required to set-up flexible, adaptive, and scalable networks. In the last years, SI has been applied to autonomic networking systems [16, 17] demonstrating its feasibility and efficiency in adapting to highly dynamic distributed systems. In analogy to the biological process of pheromone emission, in [6] each node sends a signalling routing packet containing its pheromone level and updates its value based on the level of its neighbors, thus creating a pheromone gradient toward the sink. The routing task is driven by the pheromone level of the network: data packets are forwarded toward the highest pheromone density zone and reach the sink following the pheromone gradient. Any change on the network condition will be reflected by an update of the pheromone level of the involved nodes; changes on the pheromone gradient will automatically drive the routing decisions toward the new optimal solution. In this way, the network can self-organize its topology and adapt to environmental changes. Moreover, when link failures occur, the network reorganization task is accomplished by those nodes near the broken links. This results in a robust and self-organized architecture.

In the following, we describe a SI based algorithm derived from [6]. Since our purpose is to study the gradient construction process, we will focus on the signalling component of the routing protocol avoiding details about data forwarding. We assume to have two types of node, *sinks* and *sensors*, and that the pheromone intensity is discretized into P different levels, ranging from 0 to $P-1$.

Routing paths toward the sink are established through the exchange of pheromone packets containing the pheromone level p $(0 \leq p \leq P-1)$ of each node. The gradient construction is triggered by sinks that maintain the highest level of pheromone $(P-1)$. The gradient construction protocol is described by Algorithms 1 and 2, differentiated with respect to the node type: sink or sensor.

Sink nodes, once activated, set their internal pheromone level to the highest value $p = P-1$, (Algorithm 1: line 1). Then, they periodically broadcast a pheromone message to their neighbors, encoding the intensity value p (Algorithm 1: lines 5-7). The time period is defined by the timer $T1$.

The pheromone level of a sensor node is initially set to 0 (Algorithm 2: line 1) and then it is updated following an *excitation - evaporation* process. Sensor nodes periodically broadcast a pheromone message containing their internal pheromone level. This activity is scheduled at fixed time intervals by appropriately setting the timer $T1$ (Algorithm 2: line 2). When the timer expires, a packet is sent to all the neighbors (Algorithm 2: line 12-14).

Excitation is triggered by the reception of the pheromone level from a neighbor (Algorithm 2: lines 6-10). As a consequence, the node updates its own level when a greater value is received; the new value is computed as a function of the current and the received pheromone level $update(p, p_n)$. In this paper, we use $update(p, p_n) = round((p + p_n)/2)$ (Algorithm 2: line 9).

The evaporation mechanism is triggered at the expiration of the timer $T2$ (Algorithm 2: lines 15), and it simply decreases the current value of p by one unit (Algorithm 2: lines 15-16), assuring it maintains a value greater or equal to 0.

Algorithm 1 Sink nodes

```
1: p ← P − 1
2: setTimer(T1)
3: loop
4:     e ← waitForEvent()
5:     if e = TIMER_EXPIRED then
6:         sendBroadcast(p)
7:     end if
8: end loop
```

3. The Markovian Agent model

Markovian Agents Models (MAMs) [8] represent systems as a collection of agents scattered over a geographical space. Each agent is described by a discrete-state continuous-time homogeneous Markov chain where two types of transitions may occur: *local transitions* and *induced transitions*. Local transitions are determined by internal features of the MA, whereas induced transitions occur as a consequence of the interactions with other MAs. Interactions are possible through *message exchanging*: when a local transition occurs, an MA can send a message that can be received or ignored by other MAs. The propagation and reception of messages is regulated by a *perception function* $u(\cdot)$, a function of the agent position in the space, of the message routing policy, and of the transmittance properties of the medium. The definition of the perception function is quite general, and allows several message routing strategies. In particular, the receiving agent can be aware of the state in which the agent that issued the message was, and uses this information to choose an appropriate action. The MA accepting an incoming message changes its state performing an induced transition.

Algorithm 2 Sensor nodes

```
1: p ← 0
2: setTimer(T1)
3: setTimer(T2)
4: loop
5:    e ← waitForEvent()
6:    if e = DATA_RECEIVED then
7:       p_n ← getDataReceived()
8:       if p_n > p then
9:          p ← update(p, p_n)
10:      end if
11:   else if e = TIMER_EXPIRED then
12:      t ← getTimer()
13:      if t = T1 then
14:         sendBroadcast(p)
15:      else {t = T2}
16:         p ← max(0, p − 1)
17:      end if
18:   end if
19: end loop
```

MAs are scattered over a finite geographical area $\mathcal{V}$ that can be either continuous or discrete. In case of a continuous space we have that $\mathcal{V} \subset \mathbb{R}^d$, where d is an integer number representing the dimension of the space. We denote by $\rho(\mathbf{v}) : \mathcal{V} \to \mathbb{R}^+$ the spatial density function of the agents. In particular, $\rho(\mathbf{v})$ is defined such that for every d-dimensional volume A in $\mathcal{V}$ (with $A \subseteq \mathcal{V}$) the number of agents in A is distributed according to a Poisson distribution with mean $\int_A \rho(\mathbf{v})d\mathbf{v}$. In this paper we focus on the 2-dimensional case, only, and consider $d = 2$.

To model an heterogeneous system such as a WSN with several sink and sensor nodes, we extend the MAM [8] by adding the capability to represent, in addition to several types of messages [10], also several classes of agents. Formally a *Multiple Agent Class, Multiple Message Type* Markovian Agents Model (M^3AM) is defined by the tuple:

$$M^3AM = \{\mathcal{C}, \mathcal{M}, \mathcal{V}, \mathcal{U}, \mathcal{R}\}, \tag{1}$$

where:

$\mathcal{C} = \{1 \dots C\}$ is the set of agent classes. We denote with MA^c an agent of class $c \in \mathcal{C}$.

$\mathcal{M} = \{1 \dots M\}$ is the set of message types. Each agent (independently of its class) can send or receive messages of type $m \in \mathcal{M}$.

$\mathcal{V}$ is the finite space over which Markovian Agents are spread.

$\mathcal{U} = \{u_1(\cdot) \dots u_M(\cdot)\}$ is a set of M perception functions (one for each message type).

$\mathcal{R} = \{\rho^1(\cdot) \dots \rho^C(\cdot)\}$ is a set of C agent density functions (one for each agent class).

Each agent MA^c of class c is characterized by a state space with n_c states, and it is defined by the tuple:

$$MA^c = \{\mathbf{Q^c}, \mathbf{\Lambda}^c, \mathbf{G}^c(m), \mathbf{A}^c(m), \boldsymbol{\pi}_0^c\}. \tag{2}$$

$\mathbf{Q^c} = [q_{ij}^c]$ is the $n_c \times n_c$ infinitesimal generator matrix of the CTMC that describes the local behavior of a class c agent. Its entry q_{ij}^c, with $i \neq j$, represents the transition rate from state i to state j and we define $q_{ii}^c = -\sum_{j \neq i} q_{ij}^c$.

$\mathbf{\Lambda}^c = [\lambda_i^c]$, is a vector of size n_c whose components represent the rate of *self-jumps* for a class c agent. It corresponds to the rate at which the Markov chain reenters the same state. Self-jumps allows an agent to send messages with an assigned rate while sojourning in a state.

$\mathbf{G}^c(m) = [g_{ij}^c(m)]$ is a $n_c \times n_c$ matrix describing the probability that an agent of class c generates a message of type m during a jump from state i to state j,. The elements of $\mathbf{G}^c(m)$ must respect the restriction $\sum_{m=1}^{M} g_{ij}^c(m) \leq 1, \forall c, i, j$ to ensure that during a transition an agent can generate at most one message.

$\mathbf{A}^c(m) = [a_{ij}^c(m)]$ is a $n_c \times n_c$ matrix, that describes the acceptance probability of type m messages for an agent of class c. A message is dropped with probability $a_{ii}^c(m)$, and it is accepted with probability $1 - a_{ii}^c(m)$. In the latter case, the agent immediately jumps to state j ($j \neq i$) with probability $a_{ij}^c(m)$, and $\sum_{j \neq i} a_{ij}^c(m) = 1 - a_{ii}^c(m)$, $\forall c, i, m$. This implies that rows of matrix $\mathbf{A}^c(m)$ sum to 1.

$\boldsymbol{\pi}_0^c$, is a probability vector of size n_c which represents the initial state distribution of an agent of class c.

The perception function: $u_m : \mathcal{V} \times \mathcal{C} \times \mathbb{N} \times \mathcal{V} \times \mathcal{C} \times \mathbb{N} \to \mathbb{R}^+$ is defined such that the values of $u_m(\mathbf{v}, c, i, \mathbf{v}', c', i')$ represent the probability that an agent of class c, in position $\mathbf{v}$, and in state i, perceives a message m generated by an agent of class c' in position $\mathbf{v}'$ in state i'.

3.1. Analysis

An M^3AM model can be analyzed solving a set of coupled differential equations.

Let us denote the total density of agents of class c in cell $\mathbf{v}$ with $\xi^c(\mathbf{v})$ and $\rho_i^c(t, \mathbf{v})$ the density of agents in state i in cell $\mathbf{v}$ at time t. We collect the state densities into a vector $\boldsymbol{\rho}^c(t, \mathbf{v}) = [\rho_i^c(t, \mathbf{v})]$

We assume that the total density of class c agents $\xi^c(\mathbf{v})$ remains constant over time, however it dynamically varies its distribution over the set of states of the agents. We have that:

$$\sum_{i=1}^{n_c} \rho_i^c(t, \mathbf{v}) = \xi^c(\mathbf{v}) \ , \quad \forall t \geq 0, \forall \mathbf{v}, c. \tag{3}$$

We are interested in computing the transient evolution of $\boldsymbol{\rho}^c(t, \mathbf{v})$. We start by defining $\beta_j^c(m)$ as the total rate at which messages of type m are generated by an agent of class c in state j:

$$\beta_j^c(m) = \underbrace{\lambda_j^c \, g_{jj}^c(m)}_{\text{ⓐ}} + \underbrace{\sum_{k \neq j} q_{jk}^c \, g_{jk}^c(m)}_{\text{ⓑ}} . \tag{4}$$

In (4), the term ⓐ gives the rate at which messages are emitted when the MA remains in the state j (λ_j), taking into account the probability $g_{jj}^c(m)$; similarly the term ⓑ is introduced to accumulate the rates of messages generated during state transitions, considering the transition rate q_{jk}^c and the generation probability $g_{jk}^c(m)$.

The rate $\beta_j^c(m)$ can be used to compute $\gamma_{ii}^c(t, \mathbf{v}, m)$, the total rate of messages of type m received by an agent of class c, in state i, at position $\mathbf{v}$, at time t. Let us consider an infinitesimal area $d\mathbf{v}'$. In that point of the space there are $\rho_j^{c'}(t, \mathbf{v}')d\mathbf{v}'$ class c' agents in the state j; all together they send

messages of type m at rate $\beta_j^{c'}(m)\rho_j^{c'}(t,\mathbf{v}')d\mathbf{v}'$. A class c agent in position $\mathbf{v}$ and in state i receives just a portion of messages generated from a class c' agents in state j located in $\mathbf{v}'$. The fraction of messages received is determined by the perception function $u()$. The total rate of received messages is then:

$$u_m(\mathbf{v}, c, i, \mathbf{v}', c', j)\beta_j^{c'}(m)\rho_j^{c'}(t,\mathbf{v}')d\mathbf{v}'.$$

Summing all the contributions coming from all the states and all the agent classes, and integrating over the entire area $\mathcal{V}$, the total rate of received message is obtained:

$$\gamma_{ii}^c(t,\mathbf{v},m) = \int_{\mathcal{V}} \sum_{c'=1}^{C} \sum_{j=1}^{n_{c'}} u_m(\mathbf{v}, c, i, \mathbf{v}', c', j)\beta_j^{c'}(m)\rho_j^{c'}(t,\mathbf{v}')d\mathbf{v}'. \tag{5}$$

We collect the rates (5) in a diagonal matrix $\mathbf{\Gamma}^c(t,\mathbf{v},m) = \text{diag}(\gamma_{ii}^c(t,\mathbf{v},m))$. This matrix can be used to compute $\mathbf{K}^c(t,\mathbf{v})$, the infinitesimal generator of a class c agent at position $\mathbf{v}$ at time t:

$$\mathbf{K}^c(t,\mathbf{v}) = \mathbf{Q}^c + \sum_m \mathbf{\Gamma}^c(t,\mathbf{v},m)\left[\mathbf{A}^c(m) - \mathbf{I}\right]. \tag{6}$$

The first term in the r.h.s. is the local transition rate matrix and the second term contains the rates induced by the interactions.

The evolution of the entire model can be studied by solving $\forall \mathbf{v}, c$ the following differential equations:

$$\boldsymbol{\rho}^c(0,\mathbf{v}) = \xi^c(\mathbf{v})\boldsymbol{\pi}_0^c \tag{7}$$

$$\frac{d\boldsymbol{\rho}^c(t,\mathbf{v})}{dt} = \boldsymbol{\rho}^c(t,\mathbf{v})\mathbf{K}^c(t,\mathbf{v}). \tag{8}$$

From the density of agents in each state, we can compute the probability of finding a class c agent at time t in position $\mathbf{v}$ in state i as:

$$\pi_i^c(t,\mathbf{v}) = \frac{\rho_i^c(t,\mathbf{v})}{\xi^c(\mathbf{v})}. \tag{9}$$

We collect all the terms in a vector $\boldsymbol{\pi}^c(t,\mathbf{v}) = [\pi_i^c(t,\mathbf{v})]$. Note that the definition of Equation (9) together with Equation (3) ensures that $\sum_i \pi_i^c(t,\mathbf{v}) = 1, \forall t, \forall \mathbf{v}$.

3.2. Solution technique

Equation (8) can be solved using conventional discretization techniques for both time and space. Volume $\mathcal{V}$ is discretized with a rectangular grid of $n_h \times n_w$ square cells of size d_s. From now on, the node location $\mathbf{v} = (h, w)$ identifies a discrete cell in position $h \in \{1, \ldots, n_h\}$ and $w \in \{1, \ldots, n_w\}$. Time is limited to an interval $[0, T_{Max}]$ and it is discretized with a uniform step Δt, yielding $T_\Delta = \lceil T_{Max}/\Delta t \rceil$ discrete time points: $t \in \{0, \Delta t, \ldots, T_\Delta \Delta t\}$. The solution is then computed using an implicit method [18]. In particular we approximate Equation (8) with:

$$\frac{\boldsymbol{\rho}^c(t+\Delta t,\mathbf{v}) - \boldsymbol{\rho}^c(t,\mathbf{v})}{\Delta t} \approx \boldsymbol{\rho}^c(t+\Delta t,\mathbf{v})\mathbf{K}^c(t+\Delta t,\mathbf{v}). \tag{10}$$

Multiplying by Δt and reordering terms we obtain:

$$\boldsymbol{\rho}^c(t+\Delta t,\mathbf{v})\,[\mathbf{I}-\mathbf{K}^c(t+\Delta t,\mathbf{v})\Delta t]=\boldsymbol{\rho}^c(t,\mathbf{v}). \tag{11}$$

Assuming at $t=0$ the initial condition (7), the solution vector $\boldsymbol{\rho}^c(t+\Delta t,\mathbf{v})$ is computed from (11) starting from $\boldsymbol{\rho}^c(t,\mathbf{v})$. The method is implicit since matrix $\mathbf{K}^c(t+\Delta t,\mathbf{v})$ depends on $\boldsymbol{\rho}^c(t+\Delta t,\mathbf{v})$ itself. Thus Equation (11) is solved at every time step applying a fixed-point iteration algorithm. The number of required iteration is however very limited since $\boldsymbol{\rho}^c(t+\Delta t,\mathbf{v})\approx\boldsymbol{\rho}^c(t,\mathbf{v})$.

Equation (10) is solved for every cell, and the continuous integral in (5) is replaced by a summation over all the $n_h\times n_w$ cells. If we denote by $n_{c*}=Max\{n_c\}$ the largest state space dimension of the c classes of MAs, the time complexity of the solution algorithm turns out to be $O(T_\Delta(n_h\times n_w)^2C^2Mn_{c*}^2)$ since: equation (8) iterates $T_\Delta C(n_h\times n_w)$ times the computation of the $\mathbf{K}^c(t,\mathbf{v})$ matrix which in turns requires M computations of the $\mathbf{\Gamma}^c(t,\mathbf{v},m)$ matrix whose complexity is $O((n_h\times n_w)Cn_{c*}^2)$. The computation of $\mathbf{K}^c(t,\mathbf{v})$ has a complexity of $O(M(n_h\times n_w)Cn_{c*}^2)$ and represents the most expensive step in the procedure, because it considers all the possible interactions among agents, messages, in every possible position. However, in most practical applications, the definition of the perception function confines the interaction of each MA to a limited number of neighboring MAs, significantly reducing the complexity of this step.

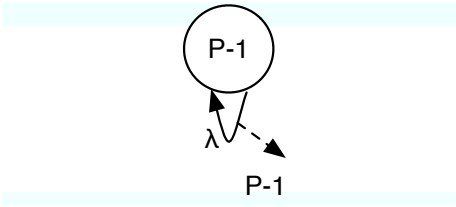
The storage complexity is limited by virtue of the iterative numerical technique in (11) that allows to compute all the needed quantities (vectors and matrices) at each iteration step. The size of the final computed vector $\boldsymbol{\rho}^c(t,\mathbf{v})$ that must be stored at each iteration step (old and new value) is $O((n_h\times n_w)Cn_{c*})$. In any case the complexity of the solution algorithm is time-bound with respect to memory-bound.

4. Description of the WSN model

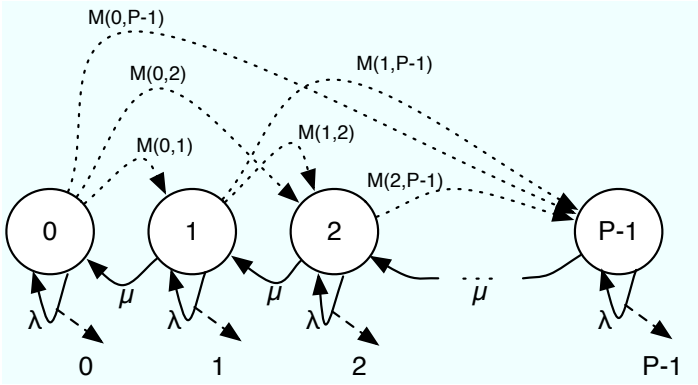
We model the SI protocol described in Section 2 with two MA classes: the class *sink node* denoted by a superscript s and depicted in Fig. 1(a) and the class *sensor node* denoted by a superscript n and depicted in Fig. 1(b). The pheromone intensity is discretized into P levels (ranging from 0 to $P-1$) that also identify the number of message types ($M=P$). We use a different message type for each possible pheromone level, and define $\mathcal{M}=\{0,1,\ldots,P-1\}$. The following pictorial rules are adopted to represent MAs. MA states are drawn as circles. Local transitions, including self-jumps, are represented by solid arrows and are labeled with the corresponding transition rate. A dashed arrow starting from a transition arc identifies the generation of the massage defined by the associate label. Induced transitions are represented with dotted arcs and are labeled with the message type that forces the transition to occur. In this work we allow only $a_{ij}^c(m)=0$ or $a_{ij}^c(m)=1$ (that is messages can only be always accepted, or always ignored), and $g_{ij}^c(m)=0$ or $g_{ij}^c(m)=1$ (that is, if messages are generated during a transition, this happens with probability 1).

The *sink* class (Fig. 1(a)) has a very simple structure, characterized by a single state. At a constant rate λ, a sink node emits a message of type $P-1$ representing the maximum pheromone intensity. The rate $\lambda=\frac{1}{T1}$ reflects the duration of timer $T1$ of the algorithms presented in Section 2.

The *sensor* class (Fig. 1(b)) has P states identifying the pheromone levels. The label inside each state indicates the corresponding pheromone intensity. In each state i $(i=0,\ldots,P-1)$ a self-loop of rate $\lambda=\frac{1}{T1}$ models the firing of timer $T1$. Message types model the current pheromone intensity of a node: we have a different message for each possible pheromone level. At each self loop transition



(a) Agent class = sink.



(b) Agent class = sensor.

Figure 1: Markovian agent models.

in a state i, a message of the corresponding type i is emitted. The evaporation phenomenon is modeled by the solid arcs (local transitions) connecting state i with state $i-1$, with $0 < i \leq P-1$. The transition rate is set to $\mu = \frac{1}{T2}$: this models the firing of timer $T2$ as described in Section 2. The key part of the algorithm is implemented in the dotted arcs (whose labels are explained in (12)) that model the transitions induced by the reception of a message. In particular, when a node in state i receives a message of type m, it immediately jumps to state j if $m \in M(i,j)$, with:

$$\begin{aligned} M(i,j) = \{m \in [0 \cdots P-1] : round((m+i)/2) = j\} \\ \forall i,j \in [0 \cdots P-1] : j > i. \end{aligned} \tag{12}$$

In other words, an MA in state i jumps to the state j that represents the pheromone level equal to the mean between the current level i and the level m encoded in the perceived message.

The N nodes (either sinks or sensors) are positioned over a uniform grid that matches the discretization structure defined in Section 3.2. Sensors can only be located in the center of each cell and we allow at most one node per cell: i.e., some cell might be empty, and $N \leq n_h \times n_w$. However, sink nodes are very few with respect to sensor nodes. Messages sent by a node are characterized by a transmission range t_r that defines the radius of the area in which an MA can perceive a message produced by another MA. This property is reflected in the perception function $u_m(\cdot)$ that, $\forall m \in [1 \cdots M]$, is defined as:

$$u_m(\mathbf{v}, c, i, \mathbf{v}', c', i') = \begin{cases} 0 & \text{dist}(\mathbf{v}, \mathbf{v}') > t_r \\ 1 & \text{dist}(\mathbf{v}, \mathbf{v}') \leq t_r, \end{cases} \tag{13}$$

where dist$(\mathbf{v}, \mathbf{v}')$ represents the distance between two nodes in position $\mathbf{v}$ and $\mathbf{v}'$. We call η the maximum number of neighbors, inside the transmission range t_r, from which an MA can perceive messages, notwithstanding border effects or node failures. In the numerical experimentation we consider two cases:

$$\begin{array}{ll} d_s \leq t_{r4} < \sqrt{2}\, d_s & \text{corresponding to} \quad \eta = 4 \\ \sqrt{2}\, d_s \leq t_{r8} < 2\, d_s & \text{corresponding to} \quad \eta = 8. \end{array} \tag{14}$$

5. Performance evaluation

In this section, we illustrate the measures evaluated in order to investigate the capability of the protocol to correctly build the routing table. The value of $\pi_i^n(t, \mathbf{v})$ provides the probability distribution of the pheromone level of the sensor node for each cell. From this raw measure, we can compute the average pheromone intensity which evaluates the forwarding attitude of the node. Since in a multi-hop gradient-based routing algorithm a node forwards its packets to the neighbor with higher forwarding attitude, we define an estimator of the gradient quality as the maximum gain in the average pheromone level. In such way, maximizing the gain corresponds to reducing the number of hops needed to reach the sink. Finally, in order to evaluate the degree of convergence of the protocol, we introduce a temporal index which defines when the average pheromone level distribution can be considered established.

The main measure of interest is the evolution of $\pi_i^n(t, \mathbf{v})$, the distribution of the pheromone intensity of a sensor node as a function of the time, over the entire area $\mathcal{V}$. $\pi_i^n(t, \mathbf{v})$ can be computed from (9) and allows us to obtain several performance indices like the average pheromone intensity $\phi(t, \mathbf{v})$ at time t for each cell $\mathbf{v} \in \mathcal{V}$:

$$\phi(t, \mathbf{v}) = \sum_{i=0}^{P-1} i \cdot \pi_i^n(t, \mathbf{v}). \tag{15}$$

The shape of the average pheromone intensity $\phi(t, \mathbf{v})$ depends on both the pheromone emission rate λ and the pheromone evaporation rate μ; furthermore, the excitation-evaporation process depends on the transmission range t_r that determines the number of neighboring cells η perceived by an MA in a given position. To take into account this physical mechanism, we define the following quantity:

$$r = \frac{\lambda \cdot \eta}{\mu}, \tag{16}$$

that gives the ratio between the global emission and evaporation process.

5.1. A measure for the gradient quality

We need to define an estimator of the gradient quality to evaluate whether the swarm-based routing algorithm correctly operates in creating a well formed pheromone distribution over $\mathcal{V}$. Our estimator is based on the fact that a multi-hop gradient-based routing algorithm forwards the packets of a node toward the neighbor with the greatest pheromone level, whether it exists, until the sink is reached. Hence, we estimate the gradient quality by maximizing the average gain in pheromone level at each hop.

To formalize this concept, we denote by $\mathcal{N}(\mathbf{v}) \subseteq \mathcal{V}$ the set of the positions of the neighboring cells that can be perceived by an MA in cell $\mathbf{v}$. Let $\langle \mathbf{v}, t \rangle$ be the MA in position $\mathbf{v}$ sending a data packet toward the sink at time t. To do the next hop, the MA $\langle \mathbf{v}, t \rangle$ selects the neighboring MA $\langle \mathbf{v}', t \rangle$ in the direction of the maximum gradient, i.e., in a position $\mathbf{v}' \in \mathcal{N}(\mathbf{v})$ with the greatest pheromone

increment, whether it exists. If such node does not exist, the data packet is not forwarded. We point out that these rules are the same used from the swarm based routing protocol in [6] to build the routing tables of sensor nodes. More formally, let $\ell(\mathbf{v}, \mathbf{v}', t) = \phi(\mathbf{v}', t) - \phi(\mathbf{v}, t)$ be the gradient value measured as the increment in the pheromone level[2] of agent $\langle \mathbf{v}', t \rangle$ with respect to agent $\langle \mathbf{v}, t \rangle$; we define:

$$\ell_{gr}(\mathbf{v}, t) = \begin{cases} \max_{\mathbf{v}' \in \mathcal{N}(\mathbf{v})} \ell(\mathbf{v}, \mathbf{v}', t) & \phi(\mathbf{v}', t) > \phi(\mathbf{v}, t) \\ 0 & otherwise, \end{cases}$$

as the maximum gradient seen by a node in position $\mathbf{v}$ at time t. The average over the space of all these values is the considered gradient quality estimator, and can be computed as:

$$\overline{\ell}_{gr}(t) = \frac{1}{N} \sum_{\mathbf{v} \in \mathcal{V}} \ell_{gr}(\mathbf{v}, t), \tag{17}$$

where N is the number of nodes in the area of interest. A high value of $\overline{\ell}_{gr}(t)$ means an high average pheromone gradient, which corresponds to a reduced number of hops.

As previously introduced, the index $\overline{\ell}_{gr}(t)$ directly arises from the routing mechanism to the purpose of finding the optimal values for the characteristic parameters of the protocol; thanks to this metrics, we will able to give a general method to set the parameters either optimizing the sensor nodes routing tables construction and the mean routing path length towards the sink node.

5.2. A measure for the time to gradient stabilization

Another practical performance index in a real WSN is the time at which the pheromone intensity distribution $\phi(t, \mathbf{v})$ can be considered established. As before, let us consider the agent $\langle \mathbf{v}, t \rangle$; we say that it is in a *stable state* when its level of pheromone $\phi(t, \mathbf{v})$ does not vary any more; since $\phi(t, \mathbf{v})$ depends on $\boldsymbol{\pi}^n(t, \mathbf{v})$, we estimate the *stable state* as the first time where:

$$\left\| \frac{\partial \boldsymbol{\pi}^n(t, \mathbf{v})}{\partial t} \right\| \leq \varepsilon. \tag{18}$$

We approximate Equation (18) with the discrete derivative:

$$\left\| \frac{\partial \boldsymbol{\pi}^n(t, \mathbf{v})}{\partial t} \right\| \approx \left\| \frac{\boldsymbol{\pi}^n(t, \mathbf{v}) - \boldsymbol{\pi}^n(t - \Delta t, \mathbf{v})}{\Delta t} \right\|, \tag{19}$$

where Δt is the discretization step, and we evaluate

$$t_s(\mathbf{v}) = \inf \left\{ t \in [0, +\infty] : \left\| \frac{\boldsymbol{\pi}^n(t, \mathbf{v}) - \boldsymbol{\pi}^n(t - \Delta t, \mathbf{v})}{\Delta t} \right\| \leq \varepsilon \right\}, \tag{20}$$

that is $t_s(\mathbf{v})$ correspond to the first time instant at which the inequality is satisfied. Since the overall network reaches the stability when all the nodes are into a stable state, the time for stability is taken as:

$$\widetilde{t} = \max_{\mathbf{v} \in \mathcal{V}} t_s(\mathbf{v}). \tag{21}$$

[2] $\ell(\mathbf{v}, \mathbf{v}', t)$ could be a negative quantity, meaning that $\langle \mathbf{v}', t \rangle$ has a lower level than $\langle \mathbf{v}, t \rangle$.

6. Model validation

We built a discrete event simulator of a WSN implementing the pheromone gradient construction algorithm, and we compared the results with the one obtained using the MAs model. A set of experiments has been conducted to compare the pheromone distribution $\phi(t, \mathbf{v})$ in the stable state conditions at time $\widetilde{t}$, with respect to different values of the r parameter.

The discrete event simulator has been implemented using the OMNeT++ [19] simulative environment. We conducted the experiments using a simple computer equipped with an Intel Core 2 Duo CPU at 2.33 GHz, 4MB L2 Cache, and 3GB RAM. Both sets of experiments have been performed considering N sensor nodes uniformly distributed over a grid of size n_h = 31 and n_w = 31, with a spatial density $\xi^n(\mathbf{v}) = 1.0\ node/cell\ \forall \mathbf{v} \in \mathcal{V}$; the discrete levels of pheromone have been assumed equal to $P = 25$. The emission time (Algorithms 1 and 2) has been fixed to $T_1 = 4.0s$, whereas we have assumed a transmission range such that $\eta = 4$. Moreover, the simulation runs have been initialized by activating each sensor node after an exponentially distributed random delay, with mean equal to $1.0\ s$. The random activation of sensors reflects the asynchronous nature of the WSN. All the measures have been computed executing 500 simulation runs, using a 90% confidence level. To simplify the presentation, we have not shown the confidence interval in the following pictures. This choice was motivated by the fact that intervals were very tight, and visually not distinguishable from their mean.

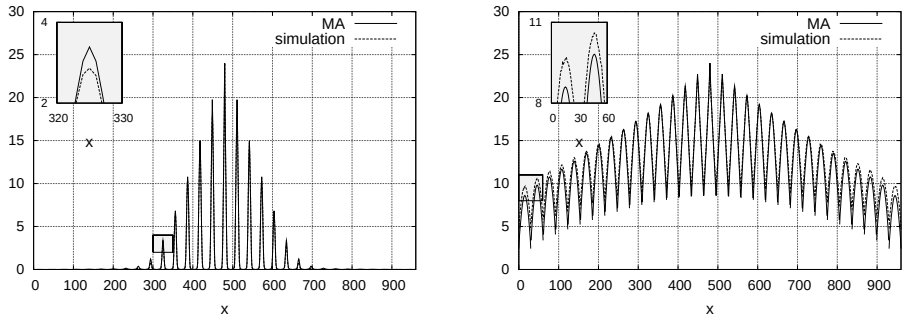
The set of experiments has been conducted assuming the values 1.0, 2.0, and 4.0 for the parameter r; the value of evaporation time T_2 has been set to $0.0625\ s$, $0.125\ s$ and $0.25\ s$, respectively. As will be explained later, such values produce different behaviors of the algorithm and they have been chosen in order to validate the model under different conditions. The time $\widetilde{t}$ to perform the transient simulation has been fixed to $40\ s$. Fig. 2 shows the pheromone distribution values $\phi(\widetilde{t}, \mathbf{v})$ obtained in the three experiments for each cell, numbered according to the following formula:

$$x(\mathbf{v}) = i * n_w + j, \tag{22}$$

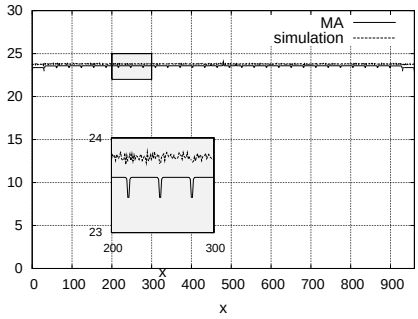
where $\mathbf{v} = (i, j)$, with $i, j = 0, \cdots, 30$.

It can be noticed that the maximum value of pheromone intensity is reached at the center of the grid where the sink is located. Since the sink coordinates are $i = j = 15$, this corresponds position $x = 15*31+15 = 480$. Departing from this location the pheromone intensity decreases. The characteristic spike-shape trend is due to the way the cells are numbered (see (22)). Each spike represents a horizontal slice of the grid. When the evaporation rate is small ($r = 1.0$), the lines in Fig. 2(a) are overlapped, meaning that the analytical and simulative results agree in the whole grid; increasing the evaporation rate (Fig. 2(b)) some discrepancies between the results can be noticed near the border of the grid, as well as in the case of saturation (Fig. 2(c)). We also depicted a magnified region of the graphs in Fig. 2 to emphasize the differences between the two results, since they were too small to be noticed in the complete plots. The differences in the final results seems to be due to the fact that the MA model uses stochastic timers, while the simulator uses deterministic clocks. However, as can be seen, discrepancies are minimal and the MA model is able to capture the overall behavior of the system.

Finally, as can be seen in Fig. 3, the computation of the analytical solution is always faster, despite the restricted number of simulation run; the results of Fig. 3 has been executed by computing a transitory to time $20\ s$. We are currently trying to exploit the possibility of aggregating the number of MAs in each cell ($\xi^n(\mathbf{v}) > 1.0\ node/cell$) to further reduce the model computational time.



(a) r=1.0 (b) r=2.0



(c) r=4.0

Figure 2: Comparison between simulative and analytical pheromone intensity distribution over $\mathcal{V}$ in the stable state.

7. Numerical experiments

The indices defined in Equation (17) and (21) have been computed under different conditions to test the SI algorithm and to provide insights into the settings of the algorithm parameters. As a first experiment we consider the dependency of the pheromone intensity $\phi(t, \mathbf{v})$ on r. We show the results obtained by solving a model defined on a square grid $\mathcal{V}$ of size $n_h = n_w = 31$ with $N = 961$ cells. We position one single sink in the center of the area (position $\mathbf{v} = (15, 15)$) and one sensor node per cell ($\xi^n(\mathbf{v}) = 1.0\ node/cell, \forall \mathbf{v} \in \mathcal{V}$) corresponding to 961 sensors. We fix $\lambda = 4.0s^{-1}$, $P = 25$, and $t_r = t_{r4}$ (i.e., $\eta = 4$ in (14)). The numerical solution is computed with $\Delta t = 0.01s$ and $\varepsilon = 0.005$.

7.1. Gradient evolution

Fig. 4 shows the pheromone distribution over $\mathcal{V}$ measured in the stable state, $\phi(\widetilde{t}, \mathbf{v})$, for different values of r. To improve the graph readability, each map is plotted both in 3D and 2D views. It can be noticed that the parameter r has a direct impact on the shape of the pheromone intensity. In particular, if r is too small ($r = 1.0$) or too high ($r = 4.0$), the quality of the gradient is poor. In fact, small r corresponds to a high value of the evaporation rate (Fig. 4(a)) that prevents diffusion of the pheromone signals, thus reducing the area covered by the sink. On the other hand, large r corresponds

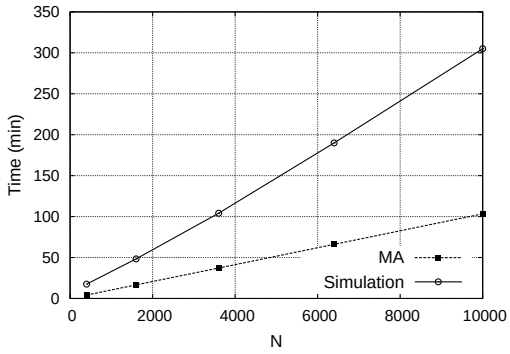
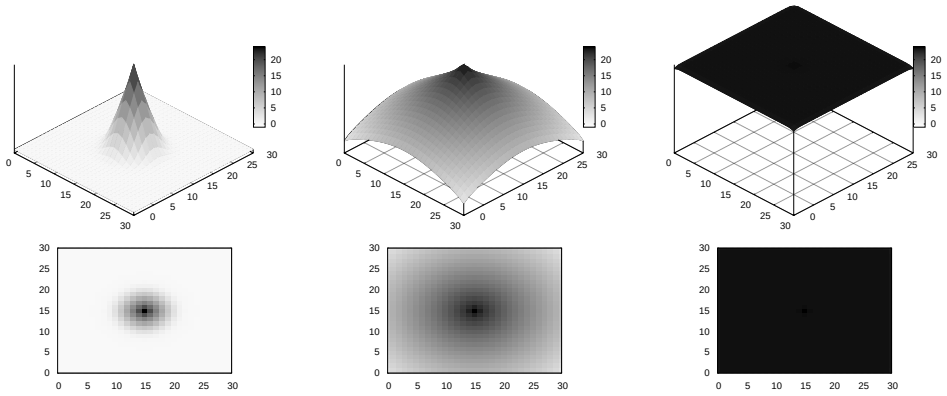


Figure 3: Comparison between simulative and analytical computing times (500 runs).



(a) r=1.0. (b) r=2.0. (c) r=4.0.

Figure 4: Distribution of the pheromone intensity $\phi(t, \mathbf{v})$ in $\mathcal{V}$ for different values of r.

to a low value of the evaporation rate (Fig. 4(c)) giving rise to a saturation of the pheromone level, thus hindering the formation of a useful gradient. Intermediate values, although giving rise to different density shapes, generate well formed pheromone gradients able to cover the whole area.

To provide a formal validation of the pheromone gradient construction process, we compute the gradient quality estimator $\overline{\ell}_{gr}(t)$ as a function of t for different values of r. Fig. 5 shows that, for low values of r (curve $r = 1.0$), $\overline{\ell}_{gr}(t)$ exhibits a monotonic behavior while for high values of r (curve $r = 4.0$) the value of $\overline{\ell}_{gr}(t)$ exhibits a maximum and then decreases due to the saturation phenomenon.

Another parameter that influences the pheromone gradient is the message transmission range t_r. Figure 6 reports the value of $(\overline{\ell}_{gr}(\widetilde{t}))$ as a function of r, varying from 1.0 to 2.4, for both $t_r = t_{r4}$ and $t_r = t_{r8}$. While in the first case we have at most $\eta = 4$ neighbors, in the second we have $\eta = 8$. In both cases we are interested in finding the value r^* of r that optimize the gradient quality $\overline{\ell}_{gr}(\widetilde{t})$ when the stabilization time $\widetilde{t}$ is reached. Observing the curve $\eta = 4$, it is possible to identify the maximum value of $\overline{\ell}_{gr}(\widetilde{t})$ at r^* = 1.8. Curve $\eta = 8$ shows a similar qualitative trend but with an overall higher value of $\overline{\ell}_{gr}(\widetilde{t})$ and with the maximum at $r^* = 1.7$. The better gradient quality can be explained by

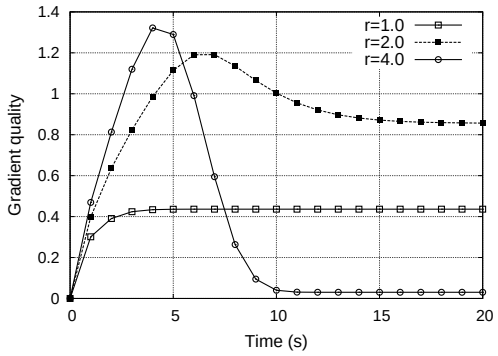


Figure 5: Transient behavior of the gradient quality estimator $\overline{\ell}_{gr}(t)$ for different values of r.

pointing out that a greater value of t_r implies a reduced number of hops needed to reach the sink with a corresponding greater value of the pheromone increment at each hop.

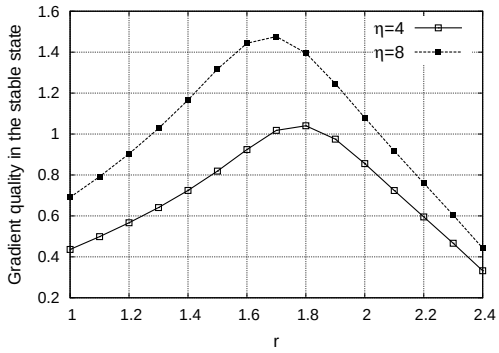


Figure 6: Gradient quality estimator in the stable condition $\overline{\ell}_{gr}(\widetilde{t})$ with respect to r varying η.

7.2. The influence of the emission rate

Once the parameter r has been set, the pheromone gradient quality is not influenced by the absolute values of pheromone emission and evaporation rates. However, changing λ influences other system properties, such as the gradient time to stability $\widetilde{t}$ and the energy spent in setting a stable gradient. To set the proper value Fig. 7(a) plots $\widetilde{t}$ as a function of λ at $r = r^*$, with both $\eta = 4$ and $\eta = 8$. It can be observed that increasing either λ or η, shorter values of $\widetilde{t}$ are obtained due to a faster and more efficient propagation of the pheromone signals in the network. In particular: *i)* An increase of λ results in higher frequency at which pheromone messages are circulated in the network, allowing to reach a stable pheromone distribution faster; *ii)* An increase of the transmission range increments the number of neighbor nodes that are able to receive a message, thus producing a more pervasive signal dissemination. Note that the decrease in the stabilization time is not simply inversely proportional to

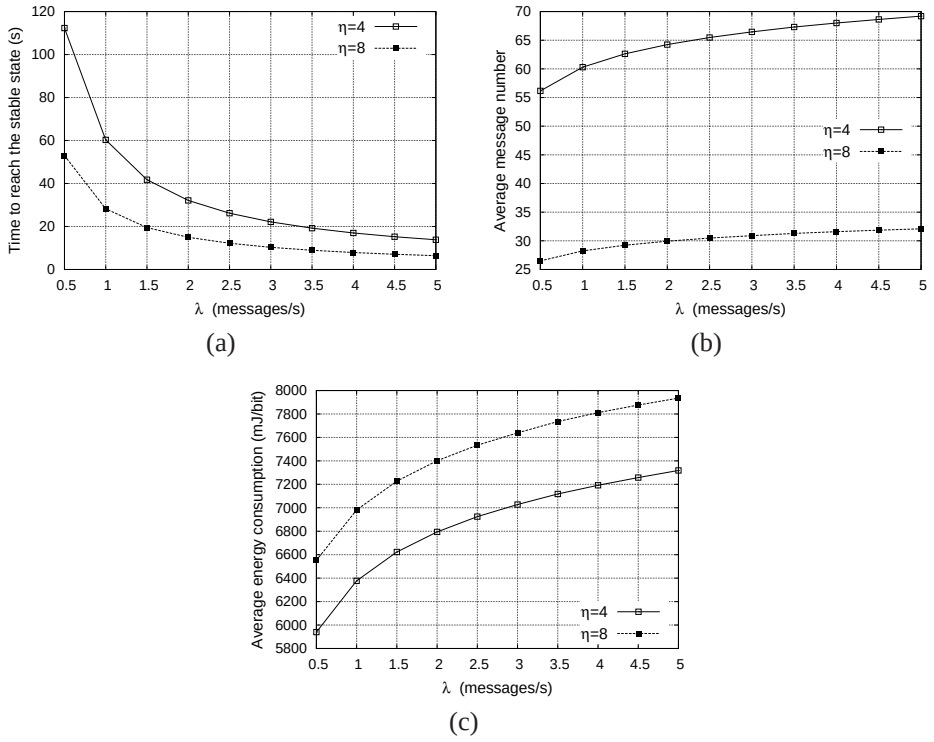


Figure 7: (a) Time to reach the stable condition $\widetilde{t}$ versus λ varying η, (b) average number of signalling messages per node $\overline{m}$, and (c) average energy consumption per node ($\overline{e}$).

the message rate (as it could be argued) because as the transmission rate increases the convergence criterion starts playing a significative role in the determination of the stopping condition.

An important aspect related to the time to reach the stable state is the average number $\overline{m}$ of signalling messages sent by each node. We can define such parameter as a function of the pheromone emission rate as:

$$\overline{m} = \lambda \cdot \widetilde{t}. \tag{23}$$

Fig. 7(b) shows an increasing trend of $\overline{m}$ as a function of λ for both values of the considered neighbor number. Increasing η from 4 to 8, the average number of signals decreases due to the larger action range of each message.

Both results in Figs. 7(a) and 7(b) tend to show that enlarging the transmission range provides better results. However, to complete the analysis, we need to take into account also energy consumptions aspects, in the computation of the cost needed to reach the stable condition. As described in [20], the energy cost per bit $E(t_r)$ required to exchange a message between a node and its η neighbors can be computed as:

$$E(t_r) = \underbrace{C_d \cdot t_r^{\alpha} + E^{(ele)} + E^{(proc)}}_{\text{ⓐ}} + \underbrace{\eta(E^{(ele)} + E^{(proc)})}_{\text{ⓑ}}, \tag{24}$$

where $E^{(ele)}$ and $E^{(proc)}$ are the consumptions due to the transceiver electronics and the processing functions, C_d is a constant factor, t_r is the transmission range needed to cover the distance between the sender and the receivers, as defined in Section 3.1, and α is the exponential power decay factor. In Equation (24) the term ⓐ refers to the energy required to send a message, while the term ⓑ corresponds to the energy consumed by the nodes receiving the message. Assuming to have a regular grid where nodes are uniformly distributed, the dependence of η as a function of t_r and d_s is given in (14). The average energy cost per node needed to reach the stable state can then be expressed as:

$$\overline{e} = E(t_r) * \overline{m}. \tag{25}$$

In agreement with [21], we use the following values for setting the parameters in (24): $E^{(ele)} = E^{(proc)} = 0.15mJ/bit$, $C_d = 0.018mJ/(bit \cdot m^{\alpha})$, and $\alpha = 2.5$. Moreover, assuming to have a square area of $900 \times 900m^2$ we set $d_s = 30m$, $t_{r4} = 32m$, and $t_{r8} = 45m$. In Fig. 7(c) we observe that, as expected, the average energy consumption needed to reach the stable state increases with λ. However, a comparison between Figs. 7(b) and 7(c) shows that, notwithstanding the lower value of messages needed, when η = 8 we obtain a greater energy cost than the case η = 4. Using the proposed model, it is then possible to estimate the cost associated to the time needed to reach the stable state and, in order to respond to particular application-specific requirements, a trade-off between $\widetilde{t}$ and $\overline{e}$ can be appropriately found during the setting phase of the network.

In order to test the model in more complex scenarios, in Fig. 8 we consider a larger network where $N = 10,000$ sensors are uniformly distributed on a grid of $3000 \times 3000m^2$ (that under the above conditions corresponds to n_h = 100 and n_w = 100) with 50 sinks placed in random locations. Using the proposed methodology, we are able to calculate the value r^* of r that maximizes the gradient quality estimator $\overline{\ell}_{gr}(\widetilde{t})$, that for the network topology shown in Fig. 8 is equal to 1.2. We can assess, with the results obtained, that the pheromone gradient is reached also when no symmetries are present in the network. Such scenario also demonstrates the scalability of the proposed analytical technique, which can be easily adopted for the analysis of extremely large networks.

7.3. Irregular topologies

Next experiments aim at analyzing the pheromone gradient construction process in the presence of irregular network topologies. When not otherwise expressed, we always refer to a value of η equal to 4. First of all, the robustness of the algorithm for the formation of the pheromone gradient is analyzed in scenarios where some sensor nodes are removed from the network (either because dormant or failed), as shown in Fig. 9.

We consider two different situations. In the first one (Fig. 9(a)), blocks of contiguous nodes are removed from the network, reproducing scenarios where nodes fail due to conditions strictly related to the geographic position. In the second one (Figs. 9(b) and 9(c)), nodes are removed randomly with an assigned percentage. Vacant cells are represented as white spots in the graphs.

From the inspection of Fig. 9(a), we observe that the algorithm is able to recognize and isolate the vacant blocks and build up a pheromone gradient that circumvents the taboo zones by creating useful paths to the sink along increasing gradient lines also for sensors that are not in direct view of the sink. The same happens for the irregular topologies of Figs. 9(b) and 9(c). However, increasing the number of removed nodes from the network (Fig. 9(c)), the gradient quality deteriorates, since some active sensors may become masked by the failed sensors and are not able to receive the pheromone messages emitted by the sink. This phenomenon is highlighted in Fig. 10 where the same topologies of Figs. 9(b) and 9(c) are considered. The failed nodes are marked by a gray circle.

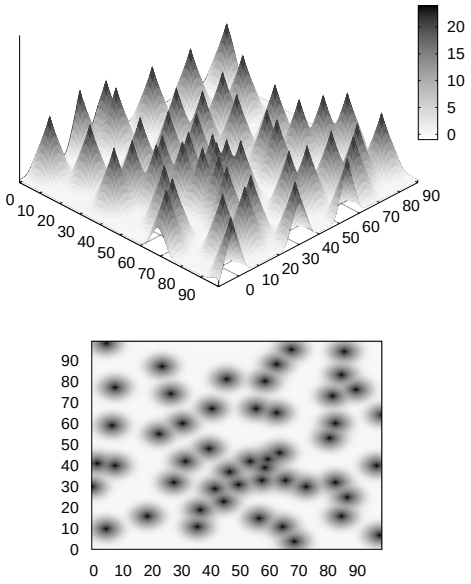
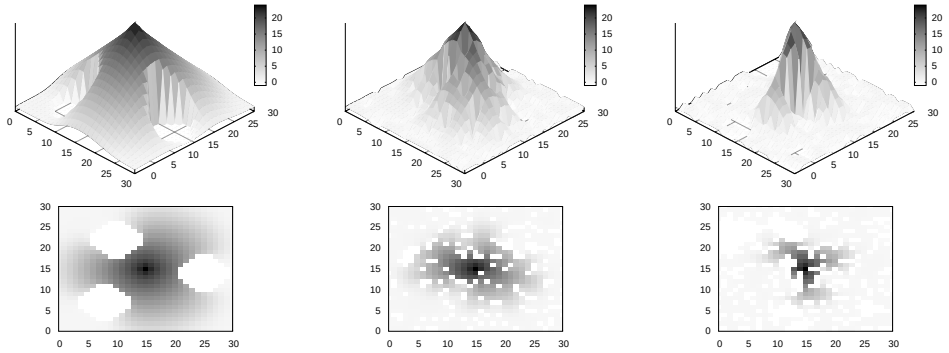


Figure 8: Distribution of the pheromone intensity over $\mathcal{V}$ in the stable condition when the network is composed by a grid of 10,000 sensor nodes with 50 sinks ($\eta = 4$, r^*=1.2).

When nodes are removed from the network (gray circles), other sensors happen to be isolated due to the absence of any path to the sink. Such unreachable nodes, marked in the graphs with the symbol x, degrade the gradient quality and their number increases as the percentage of non-active node increases. The appearance of unreachable nodes may be mitigated by increasing the transmission range t_r. As an example, compare Fig. 10(b) obtained with $t_r = t_{r4}$ (i.e., $\eta = 4$ in (14)) with Fig. 10(c) obtained with $t_r = t_{r8}$ ($\eta = 8$): the number of unreachable sensors is drastically reduced (in Fig. 10(c) only the sensor in position $(0, 30)$ remains isolated). Increasing the transmission range improves the network connectivity and efficiency but at the cost of higher power consumption and reduced network lifetime.

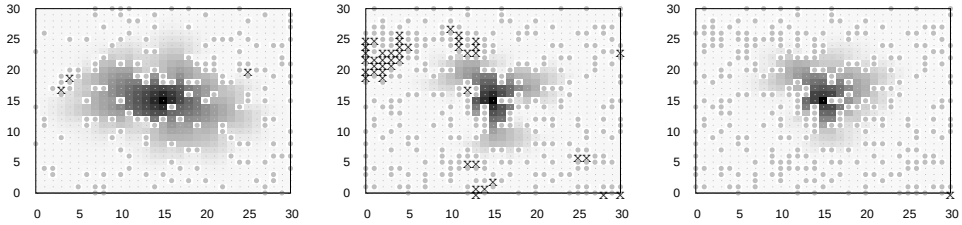
8. Related Work

In the literature, simulation is the first choice for the study of mobile and ad hoc networks [7]. In [22] and [23] the authors survey conference papers and found a number of common simulation study pitfalls such as an incorrect use of pseudo random number generators or an inadequate statistical analysis of the simulation outputs. Moreover, such studies are mostly scenario and simulator specific, therefore their results cannot be generalized to other scenarios and simulators. However, few works propose analytical models for the performance of routing algorithms. An attempt to tackle the problem analytically is in [24], but the analysis is limited to the asymptotic behavior of a two-nodes two-links system. In [25], Markov chains are used to compute the steady state routing probabilities given the routing parameters and network costs, [26] proposes a probabilistic performance evaluation framework that



(a) (b) (c)

Figure 9: Distribution of the pheromone intensity over $\mathcal{V}$ when some nodes are removed from the network ($r^* = 1.8$). (a) Contiguous areas, (b) 20% of nodes, and (c) 35% of nodes.



(a) 20% of nodes, η=4. (b) 35% of nodes, η=4. (c) 35% of nodes, η=8.

Figure 10: Removed nodes (circles), isolated nodes (X), and the pheromone intensity distribution over $\mathcal{V}$ in the stable condition.

can be used to model performance metrics such as routing overhead and energy consumption. Markov chains were used also for studying gossip protocols, in [27] gossip-based membership protocols are modeled as a Markov chain representing random membership graph transformations; stationary distribution of such chain is thus analyzed to determine the expected properties of the protocol. Also mean-field technique was proposed in [28] to approximate the behavior of a Gossiping Time Protocol implemented in a very large network and to investigate its performance. All these models ignore the spatial relationships among nodes of the network, however such characteristics are necessary, for instance in wireless network, to consider the effects of the range of the transmitter unit on the performance of the protocols. The application of stochastic geometry to the performance evaluation of communication networks [29] is a first attempt in this direction, however such models lacks the expressiveness to represent complex behavior. Instead, MA models are a modeling technique suitable to deal with systems composed by a multitude of interacting entities with complex behavior, whose spatial location is also relevant in determining their interaction.

9. Conclusions

The analytical study of the gradient formation in large WSN outlined in this paper has been carried out by resorting to an analytical model based on interacting Markovian Agents. This has allowed testing the effectiveness of the routing protocol even in contexts composed of several thousands of nodes: a size that, in general, cannot be handled with conventional discrete event simulation.

Swarm intelligence mechanisms in which the global behavior is built up starting from very short range interactions (each MA is able to interact only with its closest neighbors) have proven to be particularly suited to be analyzed using the MA technique. The analytical studies carried out on several examples illustrated in this paper have shown the adaptability of the routing algorithm to the changing conditions of the WSN in terms of parameter values and topology. The proposed model could then be exploited to effectively evaluate the parameter set in order to guarantee the convergence of the algorithm toward a stable pheromone gradient and the efficient set up of the routing tables.

Although the results obtained are encouraging, a lot of work still needs to be done to further explore both the routing algorithm and the modeling power of the MA. Future research will move towards investigating the behavior of WSN in operational conditions, assuming the routing algorithm presented in this paper, and evaluating the forwarding process with respect to the settings of the routing protocol. In particular, it is intended to develop a specific model to study the network traffic generated by an application.

To this end we need to add new types of messages representing the packets transmitted by the sensors, and to add new states to the sensor nodes to characterize the transmission, the reception, and the possible queueing of the message packets.

Modeling the transmission of messages from the sensors to the sink(s), using the paths defined by the routing algorithm presented in this paper, will allow us to study the WSN in different load conditions, with different application level strategies such as message aggregation and on-off behavior for energy saving. In this way, we should be able to provide network designers with a tool for tuning the system parameters.

Acknowledgments

This work has been partially supported by MIUR fund through the PRIN 2007 project under grant 2007J4SKYP.

References

[1] D. Bruneo, M. Scarpa, A. Bobbio, D. Cerotti, and M. Gribaudo, “Analytical modeling of swarm intelligence in wireless sensor networks through Markovian Agents,” in *VALUETOOLS09*. ICST/ACM, October 2009.

[2] I. Akyildiz, W. Su, and E. Cayirci, “A survey on sensor networks,” *IEEE Communication Magazine*, vol. 40, no. 8, pp. 102–114, August 2002.

[3] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, “Energy Conservation in Wireless Sensor Networks: a Survey,” *Ad Hoc Networks*, Vol. 7, no. 3, May 2009.

[4] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Elsevier Ad Hoc Networks*, vol. 3, pp. 325–349, 2005.

[5] M. Hinchey, R. Sterritt, and C. Rouff, "Swarms and swarm intelligence," *IEEE Computer*, pp. 111–113, April 2007.

[6] M. Paone, L. Paladina, D. Bruneo, and A. Puliafito, "A Swarm-based Routing Protocol for Wireless Sensor Networks," in *6th IEEE International Symposium on Network Computing and Applications (NCA '07)*, 2007, pp. 265–268.

[7] I. Stojmenovic, "Simulations in Wireless Sensor and Ad Hoc Networks: Matching and Advancing Models, Metrics, and Solutions," *IEEE Communications Magazine*, vol. 46, no. 12, pp. 102–107, 2008.

[8] D. Cerotti, M. Gribaudo, and A. Bobbio, "Analysis of on-off policies in sensor network using markovian agents," in *4-th Int. Workshop PerSens*, 2008, pp. 300–305.

[9] A. Bobbio, D. Cerotti, and M. Gribaudo, "Presenting dynamic markovian agents with a road tunnel application," in *MASCOTS09*. IEEE-CS, 2009.

[10] D. Cerotti, M. Gribaudo, and A. Bobbio, "Disaster Propagation in Heterogeneous Media via Markovian Agents," in *3rd International Workshop on Critical Information Infrastructures Security*, 2008.

[11] F. Zhao and L. Guibas, *Wireless sensor networks - An information processing approach*. Morgan Kaufmann, 2004.

[12] C. Schurgers and M. Srivastava, "Energy efficient routing in wireless sensor networks," in *Communications for network-centric operations: creating the information force (MILCOM 01)*, 2001.

[13] L. Lin, N. B. Shroff, and R. Srikant, "Energy-aware routing in sensor networks: A large system approach," *Elsevier Ad Hoc Networks*, vol. 5, no. 6, pp. 818–831, 2007.

[14] K. Akkaya and M. Younis, "Energy and QoS Aware Routing in Wireless Sensor Networks," *Cluster Computing*, vol. 8, pp. 179–188, 2005.

[15] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM 00) Boston*, 2000, pp. 56–67.

[16] G. D. Caro and M. Dorigo, "Ant Colonies for Adaptive Routing in Packet-Switched Communications Networks," *Lecture Notes in Computer Science - Parallel Problem Solving from Nature*, vol. 1498, pp. 673–682, 1998.

[17] X. Cui, T. Hardin, R. K. Ragade, and A. S. Elmaghraby, "A swarm-based fuzzy logic control mobile sensor network for hazardous contaminants localization," in *IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, 2004, pp. 194–203.

[18] W. Miranker, *Numerical Methods for Stiff Equations*. Dordrecht: Reidel, 1981.

[19] A. Varga, "Omnet++ - portable simulation environment in c++." Technical University of Budapest, Budapest, Hungry, Tech. Rep., 1992.

[20] C. Chiasserini, R. Gaeta, M. Garetto, M. Gribaudo, D. Manini, and M. Sereno, "Fluid models for large-scale wireless sensor networks," *Performance Evaluation*, vol. 64, no. 7-8, pp. 715 – 736, 2007.

[21] T. S. Rappaport and T. Rappaport, *Wireless Communications: Principles and Practice (2nd Edition)*. Prentice Hall PTR, December 2001.

[22] K. Pawlikowski, "Do not trust all simulation studies of telecommunications networks," in *International Conference on Information Networking (ICOIN*, 2003, pp. 899–908.

[23] S. Kurkowski, T. Camp, and M. Colagrosso, "Manet simulation studies: The incredibles," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 9, pp. 50–61, 2005.

[24] M. Roth and S. Wicker, "Asymptotic pheromone behavior in swarm intelligent manets: An analytical analysis of routing behavior," in *Sixth IFIP IEEE International Conference on Mobile and Wireless Communications Networks (MWCN)*, 2004.

[25] M. Roth, "A framework and model for soft routing: The markovian termite and other curious creatures," in *ANTS Workshop*, 2006, pp. 13–24.

[26] M. Saleem, S. A. Khayam, and M. Farooq, "A formal performance modeling framework for bio-inspired ad hoc routing protocols," in *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation*. ACM, 2008, pp. 103–110.

[27] M. Gurevich and I. Keidar, "Correctness of gossip-based membership under message loss," in *PODC '09: Proceedings of the 28th ACM symposium on Principles of distributed computing*. ACM, 2009, pp. 151–160.

[28] R. Bakhshi, L. Cloth, W. Fokkink, and B. R. Haverkort, "Meanfield analysis for the evaluation of gossip protocols," *SIGMETRICS Perform. Eval. Rev.*, vol. 36, no. 3, pp. 31–39, 2008.

[29] F. Baccelli, O. Dousse, M. Haenggi, J. G. Andrews, and M. Franceschetti, "Stochastic geometry and random graphs for the analysis and design of wireless networks." *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 7, pp. 1029–1046, 2009.