



ELSEVIER

Contents lists available at ScienceDirect

Information Sciences

journal homepage: [www.elsevier.com/locate/ins](http://www.elsevier.com/locate/ins)

## Swarm intelligence based routing protocol for wireless sensor networks: Survey and future directions

Muhammad Saleem<sup>a,\*</sup>, Gianni A. Di Caro<sup>b</sup>, Muddassar Farooq<sup>c</sup>

<sup>a</sup> Center for Advanced Studies in Engineering (CASE), 44000 Islamabad, Pakistan

<sup>b</sup> Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), 6900 Lugano, Switzerland

<sup>c</sup> nexGIN-RC, National University of Computer and Emerging Sciences (NUCES), 44000 Islamabad, Pakistan

### ARTICLE INFO

#### Article history:

Available online 23 July 2010

#### Keywords:

Wireless sensor networks  
Routing protocols  
Swarm intelligence  
Ant colony routing  
Bee-inspired routing

### ABSTRACT

Swarm intelligence is a relatively novel field. It addresses the study of the collective behaviors of systems made by many components that coordinate using decentralized controls and self-organization. A large part of the research in swarm intelligence has focused on the reverse engineering and the adaptation of collective behaviors observed in natural systems with the aim of designing effective algorithms for distributed optimization. These algorithms, like their natural systems of inspiration, show the desirable properties of being adaptive, scalable, and robust. These are key properties in the context of network routing, and in particular of routing in wireless sensor networks. Therefore, in the last decade, a number of routing protocols for wireless sensor networks have been developed according to the principles of swarm intelligence, and, in particular, taking inspiration from the foraging behaviors of ant and bee colonies. In this paper, we provide an extensive survey of these protocols. We discuss the general principles of swarm intelligence and of its application to routing. We also introduce a novel taxonomy for routing protocols in wireless sensor networks and use it to classify the surveyed protocols. We conclude the paper with a critical analysis of the status of the field, pointing out a number of fundamental issues related to the (mis) use of scientific methodology and evaluation procedures, and we identify some future research directions.

© 2010 Elsevier Inc. All rights reserved.

## 1. Introduction

Wireless sensor networks (WSNs) [3] consist of a large number of autonomous nodes equipped with sensing capabilities, wireless communication interfaces, and limited processing and energy resources. WSNs are used for distributed and cooperative sensing of physical phenomena and events of interests. Usually, the nodes are statically deployed over vast areas. However, they can also be mobile and capable of interacting with the environment. In these cases, the network is more appropriately referred to as a robotic network and/or as a sensor-actor network. WSNs can be employed in a wide spectrum of applications in both civilian and military scenarios, including environmental monitoring, surveillance for safety and security, automated health care, intelligent building control, traffic control, object tracking, etc. (see [64,83] for general overviews).

From the point of view of information processing, in WSNs, aggregation of the sensed data and its use for statistical inference can be realized in a number of ways, resulting in different network architectures. The most studied architectures [79]

\* Corresponding author. Tel.: +92 321 5063473; fax: +92 51 8314660 61.

E-mail addresses: [msaleem@case.edu.pk](mailto:msaleem@case.edu.pk) (M. Saleem), [gianni@idsia.ch](mailto:gianni@idsia.ch) (G.A. Di Caro), [muddassar.farooq@nu.edu.pk](mailto:muddassar.farooq@nu.edu.pk) (M. Farooq).

include the cases in which individual sensor nodes send their readings towards: (i) a global monitoring node (commonly indicated as a *sink*), that performs full data aggregation and inference, (ii) intermediate sink nodes, that can individually or cooperatively process the data before sending the results to the global sink or locally trigger the appropriate actions, and (iii) a global sink, but with partial data aggregation being performed on the way as data packets are forwarded hop by hop from sensor nodes towards the sink. Cases (ii)–(iii) are commonly referred to as *in-network aggregation*, and are meant to optimize energy and bandwidth resources.

In a WSN, individual nodes have limited communication range and form an ad hoc network over a shared wireless medium. Both data and control packets need to be routed in multihop modality. Data communications can be established between the nodes in the network in order to support different activities. For instance, they can be directed from a sensor node to a monitoring node for inference (which is the most common communication pattern), from a sensor/monitoring node to another sensor/monitoring node with the aim of performing some form of local cooperation, or from a monitoring node to one or more sensor nodes in order to disseminate control information. The design and implementation of *routing* schemes that are able to effectively and efficiently support information exchange and processing in WSNs is a complex task. A number of theoretical issues and practical limitations must be thoroughly taken into account. First, in order to maximize *network's lifetime*, the mechanisms adopted for route discovery and information routing need to be energy efficient. Second, since the nodes usually operate in an unattended fashion, the network is expected to display *autonomic properties* [48], meaning that the protocols in use must be self-organizing and robust to failures and losses. Last but not least, the routing protocol must be able to handle *large and dense networks*, and the associated challenges resulting from radio interference and from the need to discover, maintain, and use potentially long multihop paths.

The requirements of routing protocols for WSNs are similar to those of routing protocols for *mobile ad hoc networks* (MANETs) [69]. However, compared to MANETs, in the case of WSNs, the restrictions on energy efficiency are more compelling, nodes are usually static, and the networks are in general assumed to be much larger. Moreover, while in the case of MANETs traffic patterns strictly depend on the application and are address centric, in WSNs they are usually *data-centric* (see Section 4.5) and, mainly, are in the form of direct and reversed multicast trees rooted at monitor nodes.

So far, a large number of different routing protocols have been proposed for WSNs based on a variety of different mechanisms and optimization criteria. General overviews can be found for instance in [2,4,10,46]. In this paper, we focus on routing algorithms for WSNs that have been designed according to the principles of swarm intelligence and review the most prominent ones.

*Swarm intelligence* (SI) [9,27,47] is a relatively novel field that was originally defined as “Any attempt to design algorithms or distributed problem-solving devices inspired by the collective behavior of social insects and other animal societies” [9]. However, nowadays SI refers more generally to the study of the collective behavior of multi-component systems that coordinate using decentralized controls and self-organization. From an engineering point of view, SI emphasizes the bottom-up design of autonomous distributed systems that can show adaptive, robust, and scalable behaviors. The SI framework encompasses other popular frameworks such as *Ant Colony Optimization* (ACO) [24,22] and *Particle Swarm Optimization* (PSO) [8,47]. Most of the work in the field of SI has been and still is inspired by collective behaviors observed in natural systems such as insect societies (e.g., ACO), flocks of birds (e.g., PSO), and schools of fishes. The basic mechanisms at work in these biological systems have been reverse engineered and properly adapted to design novel algorithms for distributed optimization and control. The same process has also driven the development of the large majority of SI-based routing algorithms for WSNs. In fact, the design of most of these algorithms has been inspired by the foraging behaviors of ant colonies, and, more recently, also of bee colonies. The main rationale behind this fact lies in the observation that these insect societies, as a collective unit, do actually solve routing problems. They need to discover and establish paths that can be used by the single insects to effectively move back and forth from the nest of the colony to sources of food. These paths are the result of the synergistic interactions among a large number of relatively simple individuals that concurrently sample paths and inform others about their characteristics using a variety of communication schemes, including indirect (e.g., pheromone-mediate communication in ants) and direct (e.g., waggle dance in bees) ones. The foraging behaviors of these insect colonies are shown to be adaptive to environmental variations, robust to losses of individuals, and fully distributed and scalable.

The analogy between these biological systems and routing in networks, and in particular in WSNs, is strict. The ant/bee colony can be seen as a distributed adaptive system of smart control packets. Each of these packets makes little use of computational and energy resources to explore the network/environment. They efficiently cooperate with each other by releasing at the nodes information about the discovered paths and their estimated quality. Due to these similarities between foraging behaviors in insect societies and network routing, in the last decade, a relatively large number of SI-based routing protocols have been developed for wired networks, satellite networks, MANETs, and, more recently, WSNs. This paper is the first attempt to review and critically discuss the most prominent SI-based routing algorithms that have been developed for WSNs. This survey aims at: (i) making a wide audience aware of the existence and of the usually good performance of a number of SI-inspired WSN routing protocols, (ii) highlighting strengths and weaknesses of the proposed algorithms with respect to the central constraints and objectives of routing in WSNs, (iii) pointing out a number of methodological flaws common to many work proposing and evaluating WSN routing protocols based on SI, and (iv) identifying and proposing a scientifically sound experimental methodology and new research directions for this relatively novel research domain.

## 1.1. Organization of the paper

The rest of the paper is organized as follows. Section 2 summarizes related work. The most challenging issues to be faced when designing WSNs routing protocols are discussed in Section 3. Section 4 introduces a novel taxonomy for WSN routing protocols. In Section 5, we briefly review the foraging mechanisms at work in ant and bee colonies and discuss how they have been reverse engineered and adapted respectively in the ACO metaheuristic and in bee-inspired systems to design routing protocols for various types of networks, and, in particular, for WSNs. A general framework for SI-based routing is introduced in Section 6. In Section 7, we review a number of selected SI-based routing protocols for WSNs referring both to the taxonomy of Section 4 and to the general SI framework of Section 6. In Section 8, we critically discuss the soundness of the methodology that is commonly adopted to test and validate the reviewed SI-based routing algorithms. Conclusions and directions for future research are identified in Section 9.

## 2. Related work

### 2.1. Other surveys of SI-based routing algorithms

The first routing algorithms based on swarm intelligence concepts date back to the second half of the '90s and were designed for wired networks. Schoonderwoerd's et al. *Ant-Based Control (ABC)* [76] addressed circuit-switched telephone networks, while the *AntNet* algorithm of Di Caro and Dorigo [17,18] was meant for best-effort IP networks [18]. More precisely, both these algorithms were developed according to the principles of *Ant Colony Optimization (ACO)* [17,22,24], a popular metaheuristic for optimization. ACO derives from the reverse-engineering and the adaptation of the shortest path behavior observed in foraging ant colonies [34]. This behavior results from the combined ability of the ants of marking their paths by laying pheromone signals and, at the same time, searching the most promising foraging areas by moving towards the directions locally marked by higher pheromone intensity. The ACO principles that are at the roots of *ABC* and *AntNet* have guided, in turn, the design of a number of other SI algorithms for routing in a variety of different network environments.

A detailed discussion on the mechanisms at work in ACO-based routing, as well as an extensive overview of the characteristics and the performance of a number of different routing algorithms, and in particular of *AntNet* and of its different versions, were first compiled in the Ph.D. thesis of its different versions, were first compiled in the Ph.D. thesis of Di Caro [17].

More recently, Farooq and Di Caro [30] have presented a comprehensive review of the most prominent routing algorithms for wired networks and MANETs that have been inspired by insect societies (ants and bees). The review highlights the characteristics specific to SI-based routing protocols and shows why these characteristics make these protocols particularly suitable to deal with the challenges posed by next generation networks. In the review, the authors first define a novel taxonomy for routing algorithms that takes into account an extensive number of different aspects (e.g. deterministic vs. probabilistic decisions, global vs. local representations, single-path vs. multiple paths, etc.), and then discuss the different algorithms with respect to the new taxonomy. They also elaborate the reasons for good performance of SI-based algorithms as observed in the simulation studies. However, the authors of [30] also point out the lack of performance evaluations based on the use of real devices and testbeds. This fact makes hard to assess the effective performance of these algorithms.

The comprehensive survey of Wedde and Farooq [87] focuses on routing algorithms for wired networks. The main objective of the survey is to understand the basic design principles and the core differences existing between routing protocols proposed by researchers belonging to different communities, namely the communities of artificial intelligence, SI, and networking. In the survey, the authors discuss how the different protocols address the main challenges of routing in wired networks. This work aims to bridge relevant work of different research communities to propose novel intelligent routing solutions for future networking systems.

Sim and Sun [80] presented a review of ACO approaches for routing and load balancing in wired networks. The authors of the review made some confusion in interpreting the existing work, since they present ACO-based routing and load balancing as two different aspects, while, in more general terms, in SI-based routing they are the two faces of the same coin. In fact, SI design intrinsically favors the spreading of the data over multiple paths, automatically resulting in load balancing (see Section 4.12). The review also discusses in some depth the different mechanisms devised to avoid the situation in which the system is unable to adapt the entries of a node routing table in spite of changing network conditions (this potential problem is indicated in the ACO literature as stagnation or locked decisions).

Ren and Meng [67] have briefly surveyed some existing ACO algorithms, genetic algorithms (GAs), PSO algorithms, reaction–diffusion mechanisms, and other biologically-inspired methodologies proposed for MANETs and wired networks, and have investigated how they could be used in WSNs for routing, clustering, and security.

Iyengar et al. [45] have investigated a couple of algorithms based on genetic algorithms as well as various versions of ant-based algorithms, and have considered their use in WSNs. The review mainly focuses on algorithms developed for routing in wired networks and MANETs, while the specific characteristics of WSNs are only marginally considered.

More recently, Ducatelle et al. [26] have made an extensive survey of ACO approaches for best-effort and quality-of-service (QoS) routing in wired networks and in MANETs. The authors of the survey also discuss the relative benefits and the features of classical top-down design vs. bottom-up design, which is typical of SI approaches. Moreover, they present *Ant*

*Colony Routing* (originally introduced in [17]), a general framework in which most SI routing algorithms can be placed and that can be used to guide the design of new SI routing algorithms for large and dynamic networks.

This paper complements all these previous review works. We report a comprehensive survey and a thorough discussion of SI-inspired algorithms that have been specifically designed for routing in WSNs. To the best of our knowledge, this is the first survey addressing these issues. We expect that this work will play an important role in the future application of the concepts of SI to the development of novel state-of-the-art routing protocols for WSNs.

## 2.2. WSNs routing protocols not based on SI principles

The bulk of research in the domain of WSN routing has been done following a non SI-based approach. The number of proposed routing algorithms is quite large. Therefore, here we discuss only a few prominent protocols. For more comprehensive surveys, the interested reader is referred to [1,2,4,10,46] and the references therein.

Heinzelman et al. [40] have proposed *Sensor Protocol for Information via Negotiation (SPIN)*, which is a data-centric protocol making use of high level meta-data descriptors to perform energy efficient routing. When a sensor node detects an event, it advertises it to its neighbors by sending ADV messages. If the neighboring nodes are interested in the event, they respond with a REQ message and the data is in turn transmitted to these nodes. *SPIN* does not ensure the delivery of relevant data to all interested nodes or even to the sink node.

*Low Energy Adaptive Clustering Hierarchy (LEACH)*, proposed by Heinzelman et al. [38], is a cluster-based approach in which clusters are formed in a self-organized way. Each cluster is controlled by a cluster head. With the aim of maximizing network lifetime, the selection of the cluster head is based on the calculation of the residual energy level. Cluster heads collect data from their cluster member nodes and, after processing, communicate the results to a global sink node. *LEACH* assumes that every node can communicate directly with the sink, which is a quite unrealistic assumption in many practical situations.

Lindsey and Raghavendra [54] have proposed *Power-Efficient Gathering in Sensor Information Systems (PEGASIS)*, which avoids the assumption of direct communication and reduces the relatively large overhead of the *LEACH* protocol. In *PEGASIS*, the nodes form a chain, and each node stores in its routing table the addresses of an upstream and a downstream node. The data collection process is initiated at the far end of the chain. Each intermediate node aggregates the received data with its local data before transmitting the result to its upstream neighbor. The last node in the chain is responsible for transmitting to the sink node.

*Directed diffusion*, proposed by Intanagonwiwat et al. [44], is a popular data-centric routing protocol for WSNs. The sink node floods queries (termed “interests”) containing the attributes of the required data towards a target region. When an interest is received by a node in the specified region, it tasks its sensors to start collecting data at the prescribed rate. Sensed data is then routed back to the sink along the reverse links. *Directed diffusion* makes use of a complex algorithm for interest/data matching that puts a relatively large computational overhead on resource-constrained sensor nodes.

*Active QUery forwarding In sensoR nEtworks (ACQUIRE)* has been proposed by Sadagopan et al. [70]. *ACQUIRE* tries to resolve complex data queries according to an energy efficient scheme. When a compound query is injected into the network, the receiving node tries to resolve it locally. If the query is not fully resolved, it is forwarded to a next hop node and the process continues.

In the *Information Directed Routing (IDR)* algorithm, proposed by Liu et al. [55], the routing objective is to minimize communication cost while maximizing information gain. This latter is calculated on the basis of probabilistic models that assign to each sensor a measure of how much it is expected to contribute to the overall network task. Clearly, this measure depends on the specific task at hand. The authors of *IDR* specifically address localization and tracking tasks. *IDR* works in two phases. In the first phase, a query is injected at an entry point and routed towards a region which is estimated to have high information content. In the second phase, the response is routed to an exit node, with the response information being progressively accumulated along the return path.

Haas and Small [36] have proposed *Shared Wireless Info-station Model (SWIM)*, which targets the delivery of the sensed events to a base station as early as possible. For this purpose, after sensing an event, a node transmits the event information to its neighbors. In this way, the information about the event is rapidly spread throughout the network. As soon as one of the nodes in the vicinity of a mobile sink becomes aware of the event, it promptly delivers the related information to the sink.

Ren and Liang [68] argue that query handling mechanisms developed for conventional databases are not suitable for WSNs, due to their limited hardware resources. To this end, they have proposed the *quality-guaranteed and energy-efficient (QGEE)* algorithm, which uses confidence intervals to determine the accuracy of a query-resolution mechanism. In *QGEE*, the sensor nodes and the query are correlated through a vector space model (VSM) which is then used to design the query correlation indicator (QCI). The QCI quantifies the probability of sensor nodes to become active. Through simulations, the authors demonstrate that *QGEE* is not only resource efficient, but also results in good packet delivery ratios.

He et al. [37] have proposed a QoS-aware protocol for data aggregation in WSNs. The authors of [37] point out that existing work in the domain mainly focuses on maximizing network lifetime within a user-defined error bound. They claim that a more realistic approach should focus on providing multiple QoS requirements including error bounds as well as other aspects (e.g., network lifetime). With this motivation, they developed a new aggregation protocol which takes into account various constraints coming from the user, the characteristics of the WSN, and the sensed data. The reported results show the good accuracy, flexibility, and simplicity (in terms of control overhead) of the algorithm.

*Maximum energy welfare (MaxEW)*, introduced by Ok et al. [59], makes use of the so-called *energy-welfare metric*. This metric is based on the averaging and the equalization of the energy of the sensor nodes. Each node tries to maximize the energy welfare of its neighborhood during the routing process. As a result, the network evolves into a globally energy-efficient and balanced routing system.

Sausen et al. [75] have investigated how to perform energy efficient broadcast routing in a WSN. They have followed two strategies to accomplish this task. In the first strategy, a mechanism for *dynamic power management with scheduled switching modes (DPM-SSM)* is added on the top of an uninformed flooding protocol. In *DPM-SSM*, state transitions from active to sleep and vice versa, are dependent upon the residual energy of a node. The second strategy makes use of a connected dominating set algorithm to form a broadcast backbone. In this way, only a subset of the nodes remain active at a time. Moreover, nodes take turns to be a part of the backbone, resulting in a reduced and balanced energy consumption.

With the aim of optimizing the energy efficiency of a WSN, Marcelloni and Vecchio [57] have proposed a lossy compression technique based on the assumption that the sensor nodes generate highly correlated data samples. As a consequence, high compression ratios may be achieved. Differential pulse code modulation, with the quantization of the difference between two consecutive samples, is introduced as a way to simultaneously remove noise and compress the sampled data. Through the use of a multi objective evolutionary algorithm, the authors have evaluated the trade-off between compression performance and information loss for different combinations and values of the quantization parameters.

### 3. Design challenges for WSNs routing protocols

As mentioned in the Introduction, there are several features of WSNs that distinguish them from more traditional wireless ad hoc networks. First, WSNs have specific traffic patterns in the form of multicast (one-to-many) and converge-cast (many-to-one) trees [64] (e.g., a sensing query is sent to all sensor nodes located in a specific area, an event is detected by multiple sensor nodes that all send the relevant information back to a monitor node). Second, WSNs are usually made up of small or tiny nodes equipped with little memory, limited non-rechargeable battery, low-end processors, and small bandwidth links. As a result, WSN protocol designers face strict constraints on the use and the availability of node resources. Third, the majority of target applications for WSNs require the deployment of the sensor nodes in large numbers, ranging from thousands to millions. Hence, the scalability of the used protocols is also a major concern [3]. Fourth, individual sensor nodes can potentially generate huge amounts of data. The transmission of every data bit to a common sink node would make use of a large amount of energy, bandwidth, and processing power. Therefore, possibly redundant information need to be detected, filtered out, and/or aggregated in order to reduce the in-network traffic. In the following subsections, we take into account these specific characteristics of WSNs and we identify a list of must-to-have features for WSN routing protocols in order to allow their use in real-world applications.

#### 3.1. Minimal computational and memory requirements

Sensor nodes are typically equipped with a low-end CPU and have limited memory. For instance, Crossbow's XM2110 mote is equipped with ATmega 1281 processor running at 16 MHz [15]. Therefore, it is customary that the routing algorithm has minimal processing overhead to make its execution feasible and effective on such a low-end processor.

#### 3.2. Autonomy and self-organization

A WSN is expected to remain operational for an extended period of time. During this time, new nodes might be added to the network, while other nodes might incur in failures or exhaust their batteries, becoming unoperational. A routing protocol must be resilient to such dynamic and generally unpredictable variations and must sustain the long-term availability of essential network services [64]. Therefore, the network protocols, and the routing protocols in particular, must be empowered with self-organizing and self-management properties, in order to let the network functioning as an *autonomic system* [48].

#### 3.3. Energy efficiency

Sensor nodes are equipped with small non-rechargeable batteries (usually less than 0.5 A h and 1.2 V) [3]. Therefore, the efficient battery utilization of a sensor node is a critical aspect to support the extended operational lifetime of the individual nodes and of the whole network. A WSN routing protocol is expected to: (i) minimize the total number of transmissions involved in route discovery and data delivery, and (ii) distribute the forwarding of the data packets across multiple paths, so that all nodes can deplete their batteries at a comparable rate. This will result in the overall increase of the network lifetime.

#### 3.4. Scalability

In a wide range of WSN applications, thousands or even millions of nodes are expected to be deployed [3]. A typical example is battle field surveillance, in which the criticality and the geographical extension of the scenarios require the

deployment of large numbers of densely distributed sensors that have short communication ranges and high failure rates. Therefore, the routing protocol should be able to effectively cope with the challenges deriving from intensive radio interference, very long paths, and unpredictable failures. Moreover, it should be able to display scalable performance in face of these challenges.

### 3.5. Architecture matching the characteristics of traffic patterns

One of the features that make WSNs clearly different from other types of networks (e.g., MANETs and wired networks for local/wide area connectivity), is the structure of the traffic patterns. The most common traffic patterns present in WSNs include: *event-driven*, *query driven*, *continuous monitoring*, and some *hybrid* combination of these [2]. An event-driven traffic is triggered when a sensor node detects an event of interest (e.g., the environment temperature goes above a certain threshold). In a similar way, a user may generate a query to a set of nodes which will respond with the required data. In continuous monitoring, data packets are sent back to the monitoring node(s) at regular intervals.

The characteristics of the traffic patterns significantly affect the choice of a routing protocol. For instance, a *proactive* approach, which is based on the periodic gathering of routing information for all possible destination nodes, is suitable for continuous traffic models but might determine an excessive energy expenditure for applications that only require sporadic data transmissions. In these cases, a *reactive/on-demand* approach might be more appropriate.

### 3.6. Support for in-network data aggregation

Sensor networks can generate large amounts of locally redundant data. For instance, when a node detects that the temperature in its surroundings has exceeded a certain threshold value, it is likely that also its neighboring nodes will detect the same event. If all these sensor nodes notify the event to the monitor node, which then can aggregate the received information to assess the event with high statistical confidence. The downside of this way of proceeding lies in the excessive use of network resources. However, not every single piece of information need to be communicated to the global sink. Information from a group of neighboring nodes can be partially aggregated and processed as close as possible to its origin. In this way, it is possible to significantly reduce the number of transmissions, saving on the limited available hardware resources and reducing the negative effects due to radio interference. A good routing protocol for WSNs must be able to effectively support the setup and the use of data paths for in-network data aggregation.

## 4. Taxonomy of routing protocols

Akkaya and Younis [2] group routing protocols for WSNs into four categories: (1) *data-centric*, (2) *hierarchical*, (3) *location-based*, and (4) *QoS-aware*. Data-centric protocols do not require a globally unique ID for each sensor node, and perform multihop routing by using attribute-based naming mechanisms. Hierarchical protocols divide the network into small clusters with a representative node acting as a cluster head. Location-aware algorithms exploit the knowledge of the geographical position of a node to perform energy efficient routing. QoS-aware protocols can explicitly deal with multi-constrained requests for data transmissions. More recently, Boukerche et al. [10] have proposed a taxonomy that enlarges Akkaya and Younis's by considering six architectural categories: *attribute-based*, *flat*, *geographical*, *hierarchical*, *multipath*, and *QoS-based*. The new category flat refers to the case in which a large number of nodes collaborate together to sense the environment. The nodes are all similar and global IDs cannot be assigned to them. The category multipath includes the algorithms that compute multiple paths from sources to destinations in order to cope effectively with failing nodes.

In this paper, we propose a more comprehensive and fine-grained classification compared to those of Akkaya and Younis, and Boukerche et al. Our proposed taxonomy covers a relatively large set of features that are of interest for generic routing protocols and, more specifically, for WSNs protocols. Our classification is based on and expands those that we previously proposed in [29,30,87]. The categories included in the taxonomy are individually discussed in the subsections that follow.

### 4.1. Single path and multipath routing

Routing protocols may maintain single or multiple routes to a given destination. Single path protocols can discover one or multiple routes and then always select the best path for data transport, discarding the other paths. On the other hand, multipath routing refers to the protocols that discover, maintain, and use multiple paths to transport the sensed data. Multipath routing protocols can help in extending the network lifetime because they favor battery depletion of different nodes at a comparable rate. In the case of so-called alternate path protocols, the information about multiple paths is maintained in the routing table but is used only as a backup in case the primary path fails.

### 4.2. Reactive, proactive, and hybrid routing

In reactive/on-demand protocols, paths are searched and setup only when required. In proactive protocols, routing information for all known destinations is maintained up-to-date all the time, irrespective of whether a destination is being

selected or not for data transmission. Some protocols use a combination of both techniques and hence are called hybrid protocols (e.g., see [19,69] for more extensive discussions and examples). Usually, a proactive approach is quite energy expensive, such that it should be adopted only if the application justifies its use (see the discussion in Section 3.5).

#### 4.3. Source and next hop routing

In next hop routing, a data packet only contain the information about its final destination. At each node, the routing protocol decides the next hop using the information stored in the local routing table (e.g., [63]). In the case of source routing, the source node encapsulates all the path information in the packet datagram, such that intermediate nodes only need to read the next hop information from the datagram and forward it accordingly (e.g., [63]). Some hybrid protocols make use of both techniques. For instance, the ant packets normally used in ACO approaches, retrace a discovered path making use of source routing, while data packets are always routed according to a next hop scheme. From the one hand, the use of source routing can be very effective to reduce per packet processing requirements and to avoid loops. From the other hand, it can limit the scalability of a protocol and can incur into problems in highly dynamic networks, especially when used to route data packets rather than control packets.

#### 4.4. Flat and hierarchical routing

Flat routing protocols view the entire network as a set of nodes located on the same hierarchical level. Their job is to find a route between any arbitrary pair of nodes. Hierarchical protocols, on the other hand, divide the network into regions called zones/clusters [1]. The nodes within a cluster only need to deliver their data to the cluster head (CH). In turn, the cluster head can be part of a further level, according to some hierarchical arrangement of the nodes rooted at the final sink nodes.

#### 4.5. Data-centric and address-centric routing

Data-centric protocols do not require globally unique node IDs, while address centric protocols do. Data-centric routing is commonly used when assigning a unique ID to each node is either not feasible [44] or appropriate given the purpose and/or the size of the network. Data-centric routing, which is also indicated as content-based routing, is the common way of operating in sensor networks, global grid infrastructures, and publish/subscribe and event-notification schemes for peer-to-peer/overlay networks [28]. In data-centric routing, data packets are named using high level descriptors and queries are generated for the named data. The nodes that have the requested data only respond to these queries.

#### 4.6. Distributed and centralized routing

In centralized routing models, discovery and maintenance of routing information is controlled by a single node known as sink/base station. In the distributed approach, each node gathers/builds routing information on its own. The distributed routing model is more robust to network variations and is therefore more appropriate for dynamic and ad hoc networks. The centralized model presents a single point of failure and might be unable to follow timely the changes due to network dynamics. On the other hand, the powerful processing capabilities of the controller node can be additionally exploited to effectively execute a variety of useful processing tasks.

#### 4.7. Best-effort and QoS-aware routing

Protocols that do not provide any guarantees in terms of quality of the service delivered to the application are categorized as best-effort. Protocols that can provide to the application routing services with quality guarantees (e.g., in terms of end-to-end delay, delay jitter, available bandwidth, packet losses, etc.) are indicated as QoS-aware.

#### 4.8. Event-driven and query-based routing

This classification is based on the nature of the applications the routing protocol is serving for. In event-driven protocols, data routing starts after the detection of an event from a sensor node. For instance, an event might be triggered when the value of a monitored variable (e.g., the temperature) exceeds a certain threshold value, or after the expiration of a timer used for the periodical report from the sensors. In the case of query-based protocols, data are sent from the sensor nodes to the monitor node in response to a specific query [70]. Some protocols may support both types of applications [66].

#### 4.9. Energy-aware routing

Routing protocols that prioritize routes on the basis of an energy metric (e.g., the residual energy of the nodes on the route) are classified as energy aware. Since nodes in sensor networks have limited non-rechargeable batteries, it is customary to make an efficient utilization of the available energy if the network has to stay operational for a long time (this might not be the case for uses of the WSN that involve the acquisition of very precise information over short time periods).

#### 4.10. Loop free

If the paths used by data packets are guaranteed to have no cycles, the protocol is termed as loop-free. In order to provide this guarantee, the protocol must include explicit mechanisms to check and avoid the possible occurrence of loops. In addition to data packets, also control packets can incur in loops (e.g., during path discovery). Looping of data packets can have a strong negative impact on network performance: it reduces data throughput and/or increases packet delay, wasting at the same time bandwidth and energy resources. Looping of control packets might be less critical, but it should still be avoided for the same reasons. Loop generation is an issue for all next hop routing protocols.

#### 4.11. Fault-tolerance

Wireless sensor networks are dynamic in nature. From the one hand, nodes can fail due to hostile environment or battery outage. On the other hand, control packets can get lost due to interference or memory/processing problems. A routing protocol which is robust to topological changes and to packet losses is termed as fault-tolerant. Multipath routing is often used as a way to provide fault-tolerance.

#### 4.12. Load balancing

Load balancing refers to the mechanism in which data packets are spread in a balanced way across multiple paths from sources to destinations. A balanced traffic distribution can help to optimize network throughput and can allow all nodes to deplete their batteries at a similar rate. The use of multipath routing is a natural way to implement load balancing. However, data spreading across multiple paths must be done by minimizing path interference, since a high rate of radio collisions would nullify the positive effects derived from the use of multiple paths.

### 5. From nature to routing in computer networks

As already pointed out in the Introduction, most of the routing protocols developed according to the design principles of SI have been inspired by behaviors observed in *ant colonies* and, in much less extent, in *bee colonies*. More specifically, the foraging behaviors of these insect societies have served as a major source of inspiration to design novel routing protocols. The reason lays in the fact that, during foraging, the individuals of the colony collectively explore the environment to discover sources of food and, once found them, they setup paths between the nest and the sources of food in order to effectively transport the food back to the nest. Therefore, the collective foraging process involves distributed exploration, discovery, setting, and use of optimized routing paths in dynamic environments. These are precisely the same ingredients involved in data routing in modern networks.

Most of the SI routing algorithms inspired by ant behaviors have been developed in the context of the framework of Ant Colony Optimization (ACO) [17,22,24], an optimization metaheuristic based on the abstraction and engineering of the basic mechanisms at work in ant colony foraging. Algorithms developed according to the ACO metaheuristic have been applied to a number of different problems ranging from combinatorial optimization to distributed robotics (e.g., see [17,24] for reviews of applications).

In more general terms, ant colonies have served as an important source of inspiration for the design of novel algorithms and systems since about two decades. On the other hand, honey bees behaviors have attracted the attention of engineers and computer scientists only in more recent times, and their application now expands from networking to optimization domains [29,86].

In the following subsections, we discuss which are the main components of the ACO metaheuristic and the communication and recruitment mechanisms of honey bee colonies that have been exploited in developing state-of-the-art routing protocols for various types of networks, and for WSNs in particular.

#### 5.1. Ant colonies and general principles of ACO for routing

It has been observed that the ants in a colony can converge on moving over the shortest among different paths connecting their nest to a food source [23,34]. The main catalyst of this colony-level *shortest path behavior* is the use of a volatile chemical substance called *pheromone*: ants moving between the nest and a source of food deposit pheromone, and preferentially move towards areas of higher pheromone intensity [33]. Shorter paths can be completed quicker and visited more frequently by the ants, and will therefore be marked with higher pheromone intensity. These paths will then attract more ants, which will in turn increase the pheromone level, until there is convergence of the majority of the ants onto the shortest path. The indirect communication and coordination among the ants based on the pheromone signals released in the environment is termed *stigmergy* [9].

The basic idea behind ACO algorithms for routing [17,20] mimics this ant behavior and is based on the acquisition of routing information through a collective learning process based on *path sampling* using ant-like control packets (agents). These ant agents are generated concurrently and independently at the nodes, with the task to try out a path to an assigned

destination. An ant agent (*forward ant*) going from source  $s$  to destination  $d$  collects information about the quality of the path (e.g. end-to-end delay and number of hops), and, retracing its way back from  $d$  to  $s$  (*backward ant*), it uses this information to update the routing tables at intermediate nodes. The routing tables, called *pheromone tables*, contain for each destination a vector of real-valued entries, one for each known neighbor node. These entries, the pheromone variables, are a measure of the goodness of going through that neighbor on the way to the destination. They are continually updated according to the quality of the paths sampled by the ants. The repeated and concurrent generation of ant agents results in the availability of *multiple paths* at each node, each with an estimated measure of quality. In turn, the ants use the pheromone tables to find their way to their destinations: at each node they choose a next hop according to a *stochastic rule*, giving higher probability to those next hops which are associated with the higher values of the combination of pheromone values and of a local heuristic function. Pheromone values are the results of the long-term collective learning of good paths from ants' actions, while the heuristic values reflect a node local situation. For instance, in *AntNet* (see Section 2), the heuristic function assigns higher values to the next hops that at the decision time have shorter link queues in terms of packets to be sent. In *AntNet*, heuristic and pheromone values are combined according to an additive function. That is, the probability  $p_{kn}^d$  to select at node  $k$  neighbor  $n$  as next hop for ants with destination  $d$  is calculated, for each  $n \in \mathcal{N}(k) = \{\text{neighbors of } k\}$ , as<sup>1</sup>:

$$p_{kn}^d = \frac{\tau_{kn}^d + \omega \eta_{kn}}{C}, \quad \text{with } \eta_{kn} = 1 - \frac{q_n}{\sum_{i=1}^{|\mathcal{N}(k)|} q_i}, \quad \omega \in [0, 1], \quad (1)$$

where the  $\tau_{kn}^d$  are the pheromone variables, representing at node  $k$  the estimated quality of selecting neighbor  $n$  as next hop for packets with destination  $d$ ; similarly,  $\eta_{kn}$  are the heuristic variables, which do not depend on the specific destination, but give a measure of the expected waiting time on each link  $k \rightarrow n$  based on  $q_n$ , the current length (in bits) of the link queue to neighbor  $n$ . Both the pheromone and the heuristic variables are dimensionless values normalized in  $[0, 1]$ .  $C$  is a normalization factor to let  $p_{nd}$  be in  $[0, 1]$ .  $C$  is calculated using the values of the  $\tau$  and  $\eta$  variables for all neighbors. The value of the parameter  $\omega$  balances at decision time the relative weight of pheromone vs. heuristic values.

Another popular way to combine pheromone and heuristic values is by a multiplicative function:

$$p_{kn}^d = \frac{[\tau_{kn}^d]^\alpha [\eta_{kn}]^\beta}{\sum_{i \in \mathcal{N}(k)} [\tau_{ki}^d]^\alpha [\eta_{ki}]^\beta}, \quad \alpha, \beta \in \mathbb{R}^+. \quad (2)$$

Data packets are routed according to a stochastic rule similar to that for ant routing. However, since exploration is the task of ants and not of data, path selection is more greedy, to avoid low-quality paths. This way, data flows are adaptively spread over *multiple paths* with a strong preference for the best paths, automatically resulting in *load balancing*.

If enough ants are sent to the different destinations, nodes can keep up-to-date information about the best paths, and automatically adapt their data load spreading.

The ant generation rates, together with the mechanisms and the parameters regulating the way pheromone tables are updated, define the network-level dynamics for the adaptive learning of the routing decision policy. A variety of different approaches have been proposed to regulate ant generation. For instance, in wired networks ants are usually generated according to a periodic scheme, while in MANETs and WSNs various reactive, proactive, and hybrid schemes have been proposed (e.g., see [25]). Various approaches have also been followed to define updating rules for the pheromone values. For instance, in the case of the mentioned *AntNet*, if the destination of the forward agent was  $d$  and it traveled from its origin  $s$  to  $d$  through the link  $n \rightarrow f$ , the pheromone table is updated as follows when the associated backward ant arrives at  $n$  after traversing link  $f \rightarrow n$ . First, a *reinforcement* value  $r \in [0, 1]$  for the  $n \rightarrow f$  choice to go to  $d$  is calculated. Then, in the pheromone table, the entries  $\tau_{ni}^d, \forall i \in \mathcal{N}(n)$ , are changed as:

$$\tau_{ni}^d = \begin{cases} \tau_{ni}^d + r(1 - \tau_{ni}^d) & \text{if } i = f \quad [\text{Positive reinforcement}], \\ \tau_{ni}^d - r\tau_{ni}^d & \forall i \in \mathcal{N}(n), i \neq f \quad [\text{Decrease by normalization}]. \end{cases} \quad (3)$$

In this way, for the same value of  $r$ , a small pheromone value  $\tau_{nf}^d$  is increased proportionally more than a large one, favoring the quick exploitation of new good paths. All other neighbor nodes implicitly receive a negative reinforcement by normalization. The value of the reinforcement is calculated by evaluating the estimated ant traveling time  $t_{n \rightarrow d}$  vs. some statistical data, maintained in the node's routing information database (see Section 6.2), related to the observed traveling times to  $d$ . The way this estimation is carried out determines the increase in  $\tau_{nf}^d$  and, therefore, affects the rate of the adaptive learning of the system.

Another rule common to many ACO implementations is the following, where  $r$  can in general belong to  $\mathbb{R}^+$ , and  $\rho \in [0, 1]$  is the so-called *evaporation* coefficient:

<sup>1</sup> In the following of the paper, in order to make the notation lighter, we usually drop the superscript  $d$  to indicate selection probabilities and pheromone values, also considering that in a WSNs there is often only one destination, represented by the sink. However, it has to be understood that,  $\tau_{ij}$  always means  $\tau_{ij}^d$  (similarly for  $p_{ij}$ ), where the value of  $d$  should be clear from the text. In fact, the same link  $k \rightarrow n$  will have different desirability for different destinations  $d$ . This is not the case, however, for the heuristic values  $\eta_{ij}$ , which are usually related only to the characteristics of the local connection between  $i$  and  $j$ , that do not depend on  $d$ .

$$\tau_{ni}^d = \begin{cases} (1 - \rho)\tau_{ni}^d + \rho r & \text{if } i = f \quad [\text{Positive reinforcement} + \text{Evaporation}] \\ (1 - \rho)\tau_{ni}^d & \forall i \in \mathcal{N}(n), i \neq f \quad [\text{Decrease by evaporation}]. \end{cases} \quad (4)$$

Pheromone evaporation favors exploration and the progressive penalization of not often reinforced paths.

It is interesting to notice that, rewriting Eq. (3) as  $(1 - r)\tau_{ni}^d + r$  and  $(1 - r)\tau_{ni}^d$ , Eqs. (3) and (4) assume a similar form. In fact, they are both exponential moving averages. In particular, if  $r$  is a constant equal to 1 in Eq. (4), the two equations have precisely the same functional form. However,  $\rho$  and  $r$  play a quite different role.  $\rho$  is a constant, used to periodically decrease the value of all pheromone variables in the system, while  $r$  is a variable, whose value depends on the evaluation of the solution built by the ant agent.  $\rho$  favors exploration, while  $r$  guides the exploitation of the good paths. An equal behavior can be obtained from the two rules by setting  $r = 1$  in Eq. (4), and  $r = \rho = \text{Constant}$  in Eq. (3). The use of constant reinforcements has been investigated for various ACO implementations, and for *AntNet* in particular [18]. While still allowing the system to learn effective routing policies, the use of constant reinforcements provides inferior performance compared to the use of reinforcements calculated according to the quality of the sampled paths.

## 5.2. Foraging principles of honey bees

Foraging strategies of colonies of honey bees have been extensively studied [77] but their use for the design of network routing protocols is relatively new. Colonies of honey bees and ants share some common features, such as distributed food collection, individuals with limited capabilities, indirect communication between the individuals, etc. However, as the nature of the insects is different between the two types of colonies, with the foraging ants moving on the terrain and the bees flying, their foraging processes differ significantly from each other.

In general terms, during foraging, bees constantly leave the hive searching for new sources of nectar. Once they find it, they bring nectar back to the hive, and try to recruit other bees to exploit the food site found by competing with each other during the recruitment process. In the following, we briefly discuss the principal strategies adopted by the bees to deal with the different aspects involved in this general process.

- *Adaptive division of labor.* Although all bees can be seen as morphologically identical, depending upon the colony requirements and individual status, each bee can play a different role during the foraging process, and can switch role over time [77]. For instance, a foraging bee may decide to abandon a nectar site being currently exploited by the colony and can start looking for an alternative site or type of food (e.g., water instead of nectar), switching to a scout bee. Labor division also depends on age. For instance, younger bees take care of hive maintenance, while older bees do engage in foraging and defense tasks.
- *Recruitment and communication.* Foraging bees keep visiting food sources and travel back to the hive carrying nutrient on. Once in the hive the forager tries to recruit other bees to exploit the food site by announcing the quality, distance, and direction of the site through the so-called *waggle dance*, which is a form of one-to-group communication.
- *Stochastic site selection.* Foraging bees in the hive respond to the waggle dance according to a sort of probabilistic decision rule giving preference for choosing nearer food sites over distant ones. Due to probabilistic selection, a number of sites other than the best one are explored simultaneously, maintaining a balance between exploitation and exploration. Moreover, the preference for nearby sites allows foragers to perform energy efficient foraging.

These strategies, as well as the general organization of the foraging process implemented by the whole bee colony, have been thoroughly abstracted and engineered, and then used as the basis for the design of novel routing algorithms [30].

## 6. A general framework for SI-based routing

ACO-based and bee colony inspired routing protocols share similar principles and structures, but still some peculiar differences exist between them. In this section, we provide a unified view of these two facets of the SI design approach through a common modular framework in which all the different instances can be placed in. From the one hand, the purpose of this way of proceeding is to have a common reference framework to describe and compare different implementations of routing algorithms based on SI. On the other hand, we aim at defining a general architecture that can serve to guide the design of future SI algorithms for network routing. We provide general guidelines to design the components and the functioning of an SI router and to define the behavior of the control agents used to setup the needed routing paths.

The proposed framework consists of five top level modules and some additional submodules. The ensemble of these modules and submodules implements the architecture and the operations at the node router. The top level modules are: (i) *mobile agents generation and management*, (ii) *routing information database (RID)*, (iii) *agent structure*, (iv) *agent communications*, and (v) *packet forwarding*. Fig. 1 summarizes the characteristics of the different modules and their relationships. The explanation of the characteristics and functionalities of the modules, which is provided in the subsections that follow, is based on the practice that has been followed so far in ACO and bee inspired implementations for routing.

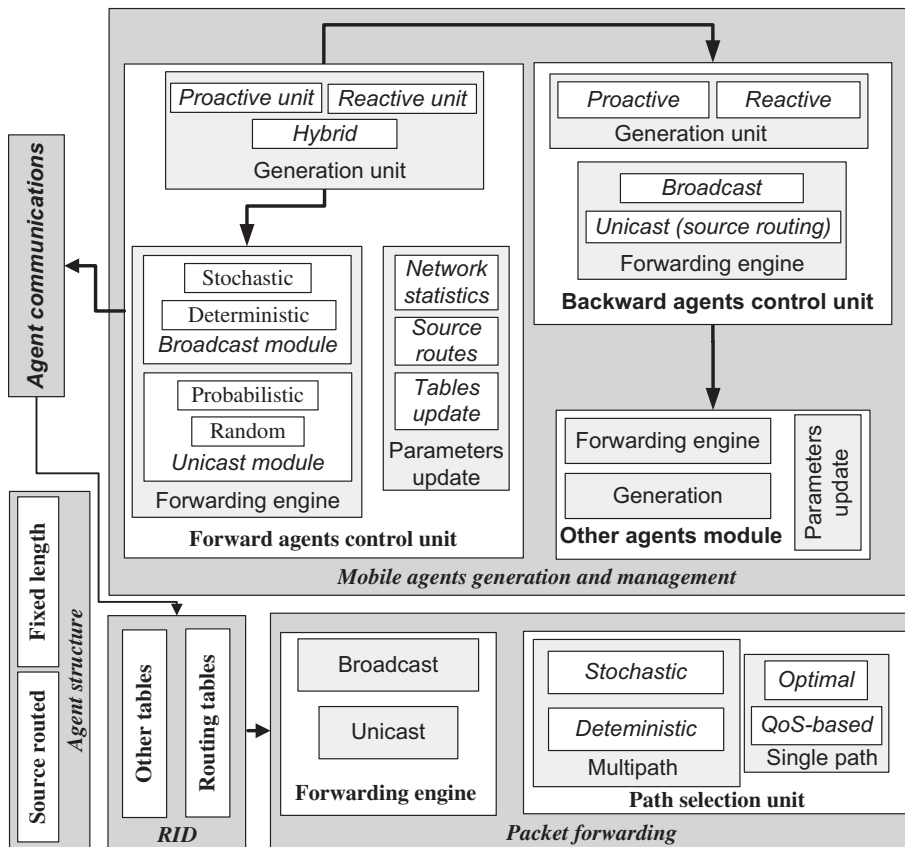


Fig. 1. Diagram of the common routing framework for SI routing protocols.

### 6.1. Mobile agent generation and management

SI-based routing protocols commonly use two types of agents to collect routing information, named respectively *forward/backward ants* in ACO-based protocols and *forward/backward scouts* in bee-inspired protocols. Forward ants/scouts are launched by source nodes to find a path to a specified destination. We call them *forward agents*. Once a forward agent reaches its destination, it travels back to the source node as a backward ant/scout. Therefore, we generically call them *backward agents*. Some protocols use other types of agents to participate in the routing process (e.g., guide ants [43] and swarms [71]).

#### 6.1.1. Forward agents control

The main duty of a forward agent is to discover a path to reach a specific destination. Additionally, on its way to the destination, it collects routing information (e.g., experienced delay, minimum remaining energy). The forward agent control module contains three components: (i) *agent generation unit*, (ii) *forwarding engine* and (iii) *parameter update module*. The agent generation unit generates forward agents according to a *proactive* (e.g., periodically), *reactive/on-demand* (e.g., following a link failure or a new route is required), or *hybrid* schedule (see [17] for more detailed discussion on reactive vs. proactive generation strategies).

Once a forward agent is generated, the forwarding engine controls its transmission from node to node. The forwarding engine either unicasts the agent to one of its neighbors or broadcasts it to all or to a selected subset of the neighbors. In most ACO-based routing schemes, next hop selection is controlled by a stochastic decision policy based on assigning the selection probabilities as a function of pheromone and heuristic values. Pheromone values are collectively learned and adapted over time by the mobile (backward) agents. At the beginning of the learning process all the next hops have equal pheromone values, such that next hop selection is mainly driven by the heuristic values.

#### 6.1.2. Backward agents control

The backward agents control module consists of a generation block and of a forwarding engine. The generation block reactively decides whether to generate or not a backward agent in response to a forward agent received at the destination (e.g., if the forward agent has found a low-quality path, it might be appropriate not to generate the backward agent). Backward agents can also be generated proactively, as is done in [49]. If the backward agents is being generated, it inherits all the

information gathered by the forward one and retraces its path back to the source node. In ACO-based schemes retracing is executed using a source-routing approach, while the next hop approach is normally used in bee-inspired protocols.

In ACO schemes, while retracing the forward path, at intermediate nodes the backward agent makes use of the routing information it carries on and of the information from the local routing information database (RID) (Section 6.2) to evaluate the quality of the forward and/or backward path. In turn, the agent communicates its evaluation and information to the agent communication module to update the RID. As a consequence, the overall selection probability of an existing path can be reinforced positively or negatively. In bee-inspired protocols, path statistics are used to compute the dance number at the source node. Dance number is a positive integer indicating the number of packets that can be sent on the path. This mimics the recruitment of foragers, the main agents that carry the data from the source to a destination, for the discovered path. High quality paths e.g., paths with high residual energy level can carry more packets and hence have a higher corresponding dance number.

### 6.1.3. Other agents control

Some implementations make use of additional agents during the path finding/updating process. For instance, it is rather common in ACO-based approaches to make use of ants that move *randomly* to explore the network and/or disseminate information in an unbiased way. Bee-inspired protocols use special agents known as *swarms* to transport the foragers back to their source nodes. In Fig. 1, we have represented these additional behaviors with a separate module which is much identical to a forward agents control block.

## 6.2. Routing information database (RID)

This is a set of locally maintained data structures. It includes the routing tables for agents and for data, as well as possible additional data structures holding statistics of interest about node and the network status that are used for path evaluation and decision making. For instance, the data concerning the expected queuing time at an outgoing link, which is used by the heuristic function in *AntNet* and other ACO approaches, are maintained in the RID. *Routing tables*, called pheromone tables in ACO, have entries of the form  $\tau_{ij}^d$ , representing the estimated goodness of selecting neighbor node  $j$  to reach destination  $d$ . The set of  $\tau$  values for the same destination can be normalized, becoming selection probabilities. Modification and update of the entries in the RID is realized through the agent communications module.

The routing information database can also serve to hold sequence numbers and other information related to passing by agents. This can be used to avoid agents carrying on the full list of visited nodes (e.g., this is the case of ACO's backward ants), that can become heavily resources-consuming or practically infeasible in very large networks [12]. Maintaining sequence numbers at the nodes also serves to avoid, during a route setup process, the multiple forwarding of an agent that has been duplicated through repeated broadcast (this is a typical problem in MANETs and WSNs protocols based on some form of flooding to find a path [19]).

### 6.3. Agent communications

The mobile agents share network data, as well as sampled and collected paths, either through direct agent-to-agent interaction, as in bee-inspired protocols, or using a stigmergic approach, as in ACO-based protocols. The agent communications module provides the logical and functional interface to mediate agent communications inside the router. It has direct access to the data in the RID and implements the rule to update routing tables and statistics. Therefore, this module is central to control the degree of adaptivity of the system. In Section 5.1, we have discussed some rules (Eqs. (3) and (4)) that have been adopted, or used as reference, in the majority of the ACO routing implementations. In case of bee-inspired protocols, the agent communications module implements a model of the bee hive dance floor, where the bee agents directly exchange their routing information. We will explain the details of this in Section 7, when we describe bee-inspired protocols.

### 6.4. Packet forwarding

This module deals with the local forwarding of data packets. It makes use of the information built by the agents and made available in the RID. The module consists of a path selection unit and of a forwarding unit. Data packets can be routed either through *multiple paths* or through the *best path*. In multipath approaches, the selection of a path among the set of available next hops is done in a *stochastic* or *deterministic* fashion either at the source node (in the case of source routing), or at the intermediate nodes (next hop routing). The stochastic approach is the most common one. As already discussed in Section 5.1, this determines the distribution of traffic across multiple paths, resulting, in WSNs, in an extended network lifetime, and, if the paths do not interfere and are comparably good, in an increased throughput. Data packets are normally unicast to their destination node. However, they can also be flooded towards the destination, as in [89]. ACO algorithms generally use next hop routing, while most of bee-inspired protocols utilize source routing.

### 7. Review of SI-inspired routing protocols for wireless sensor networks

In this section, we review selected SI routing protocols for WSNs and highlight their properties with respect to the taxonomy of Section 4. A summary of the properties for all the algorithms is shown in Table 1. In the following subsections, we first discuss ACO-based protocols and then bee-inspired protocols.

#### 7.1. Algorithms adapting AntNet to WSNs

Zhang et al. [89] have investigated the use of the AntNet [17,18] algorithm (see Section 2), originally introduced for wired networks, in WSNs, and have proposed a basic WSN routing algorithm directly derived from AntNet, and three variants of it named Sensor-driven Cost-aware Ant Routing (SC), Flooded Forward Ant Routing (FF), and Flooded Piggybacked Ant Routing (FP-Ant).

The basic AntNet-like algorithm proposed by Zhang et al. makes use of unicast forward ants generated by source (sensor) nodes. The launching interval  $I$  is set adaptively. The cost of a sampled path is calculated in terms of the number of hops. A node along the path receiving a backward ant, makes use of the sampled cost  $C$  to update RID's statistics about the observed costs from itself to the destination (a sink node) in terms of average cost  $\mu$  and variance  $\sigma^2$ . The updating is based on the same rules adopted in AntNet:

$$\mu = \mu + \eta(C - \mu), \quad \sigma^2 = \sigma^2 + \phi((C - \mu)^2 - \sigma^2), \tag{5}$$

where  $\phi$  is a constant depending on  $M$ , which is the size (in terms of number of samples) of an observation window  $W$  where the costs of the last sampled paths are stored in. The pheromone values for the destination are updated as in Eq. (3), with the reinforcement value  $r \in [0, 1]$  being calculated as in the original AntNet:

$$r = k_1 \left( \frac{C_{inf}}{C} \right) + k_2 \left( \frac{C_{sup} - C_{inf}}{(C_{sup} - C_{inf}) + (C - C_{inf})} \right), \tag{6}$$

where  $C_{inf} = \min(W)$ ,  $C_{sup} = \mu + z(\frac{\sigma}{M})$ ,  $k_1 + k_2 = 1$ , and  $z > 0$ . Clearly, the higher the cost compared to what it has been observed so far in  $W$ , the smaller the pheromone reinforcement will be for the sampled path. After receiving a backward ant at the source node, the launching interval for the forward ants is set as  $I = e^{r-0.5}I$ . In this way, if paths with better costs are discovered, the launching interval is reduced, otherwise it is increased. Data packets are stochastically distributed across multiple paths.

The three proposed variants of this basic algorithm to make it more viable for WSNs are described in the following subsections.

**Table 1**  
Features of SI-inspired routing protocols for WSNs.

Characteristics	Routing protocols								
	SC	FF	FP	EEABR	ACO-QoSR	ASAR	T-ANT	BeeSensor	SDG
Single path (S) multipath (M)	M	M	M	M	M	M	S	M	M
Reactive (R) proactive (P) hybrid (H)	H	H	H	P	R	P	P	R	P
Best effort (B) QoS (Q)	B	B	B	B	Q	Q	B	B	B
Loop free	No	No	No	No	No	No	Yes	Yes	No
Load balancing	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Fault tolerant	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Energy aware	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
Next hop (N) source routing (SR)	N, SR	N, SR	N, SR	N	N, SR	N, SR	N	N	N
Flat (F) hierarchical (HR)	F	F	F	F	F	H	H	F	H
Data centric (DC) address centric (AC)	AC	AC	AC	AC	AC	AC	AC	AC	AC
Distributed (D) centralized (C)	D	D	D	D	D	D	D	D	D
Query-based (QR) event-driven (E)	E	E	E	E	E	E	E	E	E
	MO-IAR	AntChain	PEADD	JARA	Ant-Agg.	PZSWID	E-D ANTS	DCR	
Single path (S) multipath (M)	M	S	M	S	M	S	M	S	
Reactive (R), proactive (P) hybrid (H)	P	R	R	H	P	R	R	P	
Best effort (B) QoS (Q)	B	B	B	B	B	B	Q	B	
Loop free	No	Yes	No	No	No	No	No	No	
Load balancing	Yes	Yes	Yes	No	Yes	Yes	Yes	No	
Fault tolerant	Yes	No	Yes	No	No	Yes	Yes	No	
Energy aware	No	No	Yes	No	No	Yes	Yes	No	
Next hop (N) source routing (SR)	N, SR	N	N	N, SR	N, SR	N	N, SR	N, SR	
Flat (F) Hierarchical (HR)	F	HR	F	HR	F	HR	F	F	
Data centric (DC) address centric (AC)	AC	AC	DC	AC	AC	DC	AC	DC	
Distributed (D) centralized (C)	D	C	D	D	D	D	D	D	
Query-based (QR) event-driven (E)	E	QR	QR	E	E	E, QR	E	E	

### 7.1.1. Sensor-driven Cost-aware Ant Routing (SC)

In the basic scheme, the routing performance is quite poor at the beginning: forward ants move according to a sort of random walk, such that it takes time before pheromone tables can indicate good paths. This is due to the fact that, in absence of a priori information, pheromone variables are initialized according to a random uniform distribution. In the SC variant, Zhang et al. try to overcome this problem assuming that each node can derive an initial estimate  $Q_n$  of the cost to the destination, in terms of hops, for each one of its neighbors  $n$  as next hop. Given  $Q_n$ , at a node  $k$ , the initial selection probability used by the forward ants is set to  $p_{kn}^d = \exp(C - Q_n) / \sum_{i \in \mathcal{N}^+(k)} \exp(C - Q_i)$ . The authors suggest that  $Q_n$  estimates can be obtained through geometric composition if the nodes know their own coordinates and those of the destination. Alternatively, the sink can start a flooding process, and the hop distance from each sensor node to the sink can be iteratively computed while the flooding messages spread throughout the network.

### 7.1.2. Flooded Forward Ant Routing (FF)

In the second variant, the source nodes flood the forward ants towards the sink node. *FF*, like *SC*, assumes that the forward ants are equipped with direction/location sensors. Forward ants in *FF* are flooded stochastically to reduce protocol overhead. Two methods are used to restrict the flooding process. First, an intermediate node  $i$  rebroadcasts a replica of a forward ant only if  $i$  estimates that it is closer to the destination than the node it has received the ant from. Second, node  $i$  waits for a random amount of time before forwarding the ant to the next hop. If, in the meantime, it receives the broadcast of the same replica of a forward ant from one of its neighbors, the node simply drops it.

### 7.1.3. Flooded Piggybacked Ant Routing (FP)

The key idea behind *FP* is to piggyback data packets into ant packets, with the aim of generating a higher rate of routing updates and fully exploit the multipath nature of ant forwarding. Each data packet is encapsulated in a separate ant agent (named *data ant*), which is then stochastically flooded towards the sink node and generates a path-updating backward ant at the sink. Backward ants are source routed (this is the case also for *SC* and *FF*). The flooding of both forward and data ants is controlled as in the *FF* case. The pheromone tables updated by the backward ants are used to restrict the flooding process.

The performance of the different variants has been tested considering an evader-pursuer simulated scenario in a small  $7 \times 7$  network using success rate, latency, and energy consumption as performance metrics. There is not a clear winner among the four algorithms. The basic algorithm has extremely poor success rates. *FP* has the highest success rates but it is not energy efficient. *FF* provides the shortest delays, while *SC* is the most energy efficient algorithm. Clearly further tests are needed, especially considering larger networks. In these networks, the node lists carried by forward and backward can be very large, making the protocols almost impractical, and interference can be an issue that could cause significant ant losses.

## 7.2. Energy Efficient Ant-Based Routing (EEABR)

The *Energy Efficient Ant-Based Routing* algorithm, proposed by Camilo et al. [12], is designed to extend network lifetime by reducing the communication overhead in path discovering. This is achieved by using fixed size ant agents and introducing energy and number of hops metrics in the pheromone update mechanism, allowing in this way to establish energy efficient paths. Forward ants in *EEABR* are generated on proactive basis at fixed intervals, and are unicast to next hop nodes selected according to a stochastic rule. Pheromone reinforcement  $\Delta\tau$  is computed as:

$$\Delta\tau = \frac{1}{E_0 - \left( \frac{E_{min}^k - H_d^k}{E_{avg}^k - H_d^k} \right)}, \quad (7)$$

where  $E_{avg}^k$  is the average energy of the nodes visited by forward ant  $k$ ,  $E_{min}^k$  is the minimum node energy,  $E_0$  is the initial energy level of the nodes, and  $H_d^k$  is the number of hops in the path. Making  $\Delta\tau$  a function of both path length and energy levels of the nodes would result in the discovery of short, as well as energy efficient, routes. At a node  $i$ , and for a sink  $d$ , *EEABR* uses the following pheromone updating rule:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \frac{\Delta\tau}{\phi B_d^k}, \quad (8)$$

where  $\phi$  is a scaling coefficient,  $\rho$  represents the pheromone evaporation factor and  $B_d^k$  is the number of nodes visited by the backward ant till the current node. The rationale behind Eq. (8) is to decrease the impact of  $\Delta\tau$  as the backward ant moves away from the sink node, favoring in this way an exploratory behavior at distant nodes. This is a particularly useful feature when the sink node is mobile. Neighbor selection probabilities are computed using Eq. (2), in which  $\eta_{ij} = E_0 - e_j$ , and  $e_j$  is the current energy level of next hop node  $j$ .

Simulation results show that *EEABR* performs comparatively better than other variants of it in terms of minimum remaining energy of a node, standard deviation in the battery levels, and energy efficiency.

Forward ants with fixed size are implemented by letting the ants keeping memory of only the two last visited nodes, and maintaining in the RID the full lists of the visited nodes together with the ant identifiers, in order to be easily retrieved (see

also Section 6.2). This reduces agent bandwidth utilization. On the other hand, the proactive nature of *EEABR* makes it quite demanding in terms of energy.

### 7.3. ACO-based quality-of-service routing (ACO-QoS)

*ACO-QoS*, proposed by Camilo et al. [11], is a reactive protocol that tries to cope with both strict delay requirements and the limited energy and computational resources available at the sensor nodes. The addressed problem consists in finding paths from sensor to sink nodes such that the total end-to-end delay is less than a bounding value  $D$ , while the energy residual ratio,  $ERR = E_{residual}/E_{initial}$ , is above a certain threshold value. In *ACO-QoS*, when a source/sensor node has data to send, it checks its routing table for an appropriate path. If such a path does not exist in the routing table, a probe phase to find a new route is initiated.  $m$  forward ants are used for each path probe. These ants are unicast to next hop nodes using selection probabilities calculated as in Eq. (2). In the *ACO-QoS* algorithm the local heuristic information  $\eta_{ij}$  is defined as the ratio between the residual energy of node  $j$  and the summary residual energy of all the neighbor nodes of node  $i$ :

$$\eta_{ij} = \frac{E_{residual}(j)}{\sum_{k \in N_i(k)} E_{residual}(k)}. \quad (9)$$

The  $\eta_{ij}$  are intended to favor the nodes with higher level of residual energy, in order to balance energy usage.

Forward ants carry a sequence number  $k = 1, \dots, m$ , in their header in order to sort the paths at the destination. Pheromone reinforcement for ant  $k$ ,  $\Delta\tau^k$ , is computed as:

$$\Delta\tau^k = \begin{cases} f(ERR_k^*, \Delta\tau^{k-1}) & \text{if path delay} \leq D, \\ 0 & \text{otherwise,} \end{cases} \quad (10)$$

where  $ERR_k^* = ERR_k / Hops^k$  is the normalized energy residual ratio of the path. Eq. (10) ensures that paths with large delays will not get a pheromone increment. Eq. (4) is used to update the pheromone variables at the nodes on the sampled paths. *ACO-QoS* also embeds the pheromone smoothing and bounding mechanism of *Max-Min Ant System* [84]: where  $\tau_{max}$  is a constant and  $\gamma \in [0, 1]$ .

*ACO-QoS*'s performance has been compared in simulation to *AODV* [63] and *DSDV* [62], two classical routing algorithms for MANETs, using four QoS metrics: end-to-end delay, packet delivery ratio, routing overhead, and energy residual ratio. *ACO-QoS* can satisfy the time constraints of the considered application and has higher packet delivery ratio than its competitors.

QoS routing in WSNs has not received the deserved attention of SI community and *ACO-QoS* is a reasonably good attempt in this context. On the other hand, the use of unicast ants for finding delay-constrained routes is questionable. Keeping the routing information in the header of forward ants is also an important shortcoming because it will not only result in more energy consumption, but delay guarantees will also be negatively affected, especially in large networks. Finally, the paper lacks of a detailed analysis of the characteristics and the performance of *ACO-QoS*.

### 7.4. Ant-based service-aware routing algorithm (ASAR)

*ASAR* is a QoS-aware routing protocol proposed by Sun et al. [85] for multimedia sensor networks. The authors of *ASAR* have targeted two different operating network modes having different QoS requirements: (a) the event-driven mode includes only one type of event-driven service, the R-service, that puts strict requirements in terms of both delay and reliability for event detection and notification (e.g., surveillance of elder people), (b) the query-driven mode includes two types of services, the D-service, which is based on data query, and the S-service, which is stream query. S-service is strictly intolerant to errors but more tolerant to delays (e.g., a user querying about parking information), while the QoS requirements of the D-service are the opposite (e.g., users query for real-time audio/video data).

A path with low traffic and high signal-to-noise ratio suits R-type applications, while a congested path with high signal-to-noise ratio may fulfill the QoS requirement of D-type applications. The requirement of S-type applications can be satisfied by selecting a path having low traffic and low signal-to-noise ratio. *ASAR* assumes a clustered network and its purpose is to build paths from the cluster heads to the sink node that can support the three types of applications. Sun et al. do not specify how the process of cluster formation is realized, how sensors send data to their cluster head, and how data aggregation is implemented.

A path  $l$  with a total of  $n$  links in *ASAR* is evaluated on the basis of four quality metrics: (1) cumulative queuing delay  $d_l$ , (2) minimum of all link bandwidths  $b_l$ , (3) packet loss rate  $p_l$ , and (4) cumulative energy consumption  $c_l$ . The objective is to find a QoS-aware path between a cluster head and the sink node for each one of the considered three types of services. Each cluster head iteratively launches a group of  $3m$  forward ants which are probabilistically unicast to the sink node. When all the ants arrive at destination, pheromone reinforcement for the used paths  $\Delta\tau$  is calculated as:

$$\Delta\tau^h = \sum_{k=1}^m \gamma_h^b(b_l) + \gamma_h^p(1 - p_h)^{\frac{1}{m}} + \frac{\gamma_h^d(d_{max} - d_l) + \gamma_h^c(c_{max} - c_l)}{|l_h|}, \quad (11)$$

where  $h \in \{R, D, S\}$  refers to service type,  $d_{max}$  and  $c_{max}$  are the bounds on delay and energy consumption, and  $\gamma^b, \gamma^p, \gamma^d, \gamma^c$  are weighting factors for bandwidth, packet losses, queuing delay, and energy consumption respectively. Eq. (11) is intended to meet the requirements of the three different services and calculates pheromone increments accordingly. For each visited link  $(i, j)$ , pheromone updates are calculated at the sink using the following equation, and then the appropriate backward ants are generated:

$$\tau_{ij}^h = \rho \tau_{ij}^h + \Delta \tau^h. \quad (12)$$

In ASAR, pheromone values assume discrete values between  $[0, \tau_{max}^h]$ . This accelerates convergence and reduces the number of needed pheromone updates (small values of  $\Delta \tau$  might not produce effects on the quantized  $\tau_{ij}$ ), with the consequent reduction of downstream control traffic (sink to source), that is, of generated backward ants. Selection probabilities are updated using Eq. (2). While pheromone values are computed using global information gathered by ants, the heuristic values  $\eta_{ij}^h$  are entirely based on local information:

$$\eta_{ij}^h = \gamma_h^b (b_{ij}^h) + \gamma_h^p (1 - p_{ij}^h) + \gamma_h^d (d_{max} - d_{ij}^h) + \gamma_h^c (c_{max} - c_{ij}^h). \quad (13)$$

The NS-2 simulator has been used to compare the performance of ASAR with *Directed diffusion* [44] (see Section 2.2) and Dijkstra's algorithm. Results are reported for a 20-node network considering latency, energy consumption, bandwidth, and packet loss rate as metrics. ASAR appears to provide only marginally better performance than its competitors. ASAR's energy consumption is the highest among the three protocols even though it does not include the overhead for cluster formation and data aggregation. Moreover, the algorithm relies on computationally-demanding calculations that could badly affect energy consumption in real WSNs.

### 7.5. Self-organizing data gathering for multi-sink sensor networks (SDG)

Kiri et al. [49] describe a cluster-based data gathering scheme aimed to achieve reliability and scalability in WSNs. In the paper, the authors argue that a WSN architecture with a single sink is not robust to energy depletion. In fact, once the nodes around the sink run out of energy, the sink remains isolated and the WSN become useless. They propose a multi-sink WSN in which the nodes can use an alternate sink in case of failure.

In the protocol, in order to minimize routing overhead, ant agents are only generated by sink nodes in the form of backward ants. These are broadcast by sink nodes on proactive basis. They carry a pheromone value  $\Delta \tau$  to be used for updating the pheromone at the sensor nodes. At the beginning,  $\Delta \tau$  is set by the generating sink  $s$  to a value  $\tau_{max}$ . A node  $i$  receiving the backward ant from neighbor  $j$ , stores the value of  $\Delta \tau$ , as well as the ID  $s$  of the generating sink and that of the neighbor  $j$ . This information is used to update the routing table, with  $\Delta \tau$  being used to update the benefit of using neighbor  $j$  to send data to  $s$ .  $\Delta \tau$  is first decreased as follows, and then the backward ant is further broadcast:

$$\Delta \tau = \alpha \left( 1 - \exp \left( -\beta \frac{E_r}{E_m} \right) \right) \Delta \tau, \quad (14)$$

where  $\alpha \in (0, 1)$ ,  $\beta \in \mathbb{R}^+$ , and  $E_r$  and  $E_m$  are respectively  $i$ 's residual and initial energy levels. The intuition behind (14) is to increase more the pheromone at the nodes that are near the sink and have high energy levels. The process is however based on flooding, which has a high overhead. Therefore, backward ants are generated at a very low rate, and are integrated by a local mechanism based on so-called *Hello Ants* (which is similar to the mechanism used in [25] for MANETs). Hello Ants are generated by the sensor nodes at a higher rate than backward ants, but do not get further broadcast after reception. A Hello Ant sent by sensor node  $j$  carries the ID of the sink node  $s$  to which it belongs to, the average  $\tau_j^s$  of all its pheromone values  $\tau_{jk}^s, k \in \mathcal{N}(j)$ , and the pheromone of the cluster handled by  $s$ . A node  $i$ , on receiving an Hello Ant from  $j$ , updates its pheromone entries if (and only if)  $i$  and  $j$  belong to the same sink  $s$ :

$$\tau_{ij} = \gamma \tau_{ij} + (1 - \gamma) \tau_j^s, \quad \gamma \in [0, 1]. \quad (15)$$

Sensor nodes communicate data and event information to their sink node through the usual ACO mechanism of stochastic forwarding. The probability that a node  $i$  will select  $j$  as a next hop node is given by:

$$p_{ij} = \frac{(\tau_{ij})^2}{\sum_{k \in \mathcal{N}(i)} (\tau_{ik})^2}. \quad (16)$$

Node clustering in this algorithm is inspired from eggs and larvae grouping behaviors observed in ant colonies. Ants repeatedly pick up and drop eggs according to their degree of similarity. Based on this, Kiri et al. define an additional pheromone variable, termed *cluster pheromone*, which reflects the advantage of being associated with the cluster headed by sink node  $s$ . The information exchanged through the Hello Ants is used to iteratively update cluster pheromone at the nodes, which is calculated as an average of the pheromone values  $\tau_{ij}^s$  of all the nodes in the same  $s$  cluster.

Nodes at the borders of their cluster can dynamically change cluster membership according to a probabilistic mechanism that favors clusters with higher cluster pheromone. If a sensor node detects that its sink node is not operational or is out of reach, it exploits cluster pheromone to allocate itself to another cluster.

The algorithm has been evaluated through simulations in NS-2. The main metric that has been considered is reliability, expressed in terms of event-notification rate, that is, the ratio between the number of sensor generated packets associated to the detection of an event and the effective number of the packets delivered at the sinks. Even in presence of very lossy communication channels, the algorithm is able to achieve event-notification rates of about 90%. According to the reported results, the algorithm can also successfully handle both random node failures and sink node failures. The disadvantage of this algorithm is that it consumes a significant amount of energy because of its proactive nature and hello packets exchange. The algorithm has not been compared to other existing protocols.

### 7.6. Data-centric routing in WSNs

Singh et al. [82] first introduce an offline centralized ACO algorithm for the solution of the Steiner tree problem (and provide a computational study of its performance), and then present an online distributed version of the algorithm for data-centric routing in WSNs. The network is modeled as a weighted graph in which the weight (cost) of an edge is the Euclidean distance between the two nodes. The task is then to form a minimal cost Steiner tree routed at the sink node. This can be used to route data/events from the sensor nodes to the sink. They specifically consider the case of a single sink, but the algorithm can be straightforwardly adapted to the multi-sink case by replication.

Each sensor node iteratively sends a forward ant to find a path from itself to the sink in a multicast tree. Therefore, if there are  $m$  sensors,  $m$  forward ants are generated at every generation round. The next forward ant is generated only after the associated backward ant arrives back at the sensor node. This introduces a level of synchrony in the algorithm. On average only  $m$  forward ants or  $m$  backward ants are present in the network. As usual in ACO algorithms, a forward ant maintains the list of visited nodes to ensure not to visit them again. If two forward ants meet at an intermediate node, they are merged into one single ant. This is because the paths followed by the joining ants become connected into one single sub-tree. In this way, the paths sampled by the set of forward ants arriving at the sink will constitute a Steiner tree. At node  $i$ , the next hop  $j \in \mathcal{N}(i)$ , is selected by the forward ant adopting the stochastic selection rule of Eq. (2). The pheromone variables  $\tau_{ij}$  encode the learned shortest paths in the Steiner tree. The heuristic variables  $\eta_{ij}$ , termed *potentials*, assigns low potential (higher desirability) to the nodes  $j$  that are close at the same time to the sink and to nodes that have been already visited by a forward ant in the current round of the algorithm. The rationale behind this is that moving a forward ant towards the path of another forward ant allows the merging of two paths, with the result of creating a single sub-tree to reach multiple sensor nodes.

A forward ant carries a variable  $pCost$  that indicates the partial cost contributed by the ant's path to the Steiner tree.  $pCost$  is set to zero at the source and is incremented with the value of  $dist(i,j)$  when going from node  $i$  to  $j$ . A counter  $tag_j$  is maintained at each node and is incremented each time a forward ant visits the node in the current round. If, on arrival at node  $j$ , the forward ant finds that  $tag_j \neq 0$ ,  $pCost$  is not incremented, since in this case node  $j$  has already been included in an ant's path.

When a forward ant arrives at the sink, the  $pCost$  values are added up to calculate the overall *cost* of the Steiner tree found by the forward ants in the current round. After all  $m$  forward ants have arrived at destination,  $m$  backward ants are generated with a copy of the *cost* variable. This is used to update pheromone variables along the paths found by the forward ants according to the formula of Eq. (4), where the reinforcement value is set proportionally to  $(cost)^{-1}$ . All the entries of the pheromone tables are subject to evaporation. Backward ants also carry two additional cost variables,  $ipCost$ , which is incremented by  $dist(i,j)$  when moving from  $j$  to  $i$ , and  $rCost$ , which is incremented as  $ipCost$  but is reset to zero each time the backward ant detects a branching in the Steiner tree. This is detected through the  $tag$  variables: if  $tag_i < tag_j$  there is a branching. On leaving a node, a backward ant decrements the  $tag$  variable of the node, such that at the end of the round they are reset to zero. The updating rule for the potential functions is reported in Eq. (17). Since ants preferentially move towards low potential (and high pheromone) nodes, according to (17) they move towards the nodes that are closer to the sink or the branching points:

$$\eta_{ij} = \gamma \cdot rCost + \lambda \cdot ipCost, \quad \gamma, \lambda \in [0, 1]. \quad (17)$$

The performance of the algorithm has been evaluated through simulations considering randomly generated networks of 50, 100, and 200 nodes. The algorithm shows convergence to near optimal solutions in about 1000 rounds. The proposed algorithm has been also compared to the state-of-the-art data-centric algorithm of Krishnamachari et al. [51]. In all the considered scenarios, the ACO algorithm was able to find a Steiner tree with a cost significantly better than that found by the competitor algorithm. However, more extensive tests are necessary to assess its performance.

The algorithm is an elegant solution for forming a multicast tree in sensor networks. However, it relies on some strong assumptions. For instance, the number of source and destination nodes are assumed to be known a priori. Second, the sink node generates backward ants only when it receives all the forward ants, and vice versa for the sensor nodes. It can be hard to implement this mechanism in practice due to the inherently unreliable nature of WSNs, such that an ant can easily get lost on its way. Another questionable aspect concerns the claim that it is a data-centric algorithm. This is not entirely true since forward ants make use of a source routing header of the path they follow, which in turn requires unique node IDs. This contradicts the design philosophy of data-centric routing [44].

### 7.7. Many-to-One Improved Ant Routing (MO-IAR)

MO-IAR, introduced by Ghasemaghaei et al. [32], is an extension of some previous work of the same authors [31]. The algorithm works in two phases. In the first phase, the protocol tries to establish shortest paths using ant agents. The second phase starts with the actual data routing. During this phase, a proactive congestion control mechanism is adopted to minimize packet losses. The protocol assumes that each sensor knows its location and the location of the destination (e.g., using GPS devices), as well as its neighbors and their distances (e.g., through the exchange of Hello messages).

During the first phase, each node  $i$  launches a sequence of  $n$  forward ants. Each ant is unicast to a neighbor node  $j \in \mathcal{N}(i)$  selected according to the following rule:

$$p_{ij} = A_{jd} \frac{\tau_{ij} + \beta d_{ij}}{1 + \beta(|\mathcal{N}(i)| - 1)}, \quad (18)$$

where  $A_{jd} = (\lambda D_{jd})^{-1}$  is a distance weighting factor, with  $D_{jd}$  being the estimated Euclidean distance between node  $j$  and the destination,  $\lambda, \beta \in [0, 1]$ , and  $d_{ij}$  is the heuristic value representing the cost (delay) of routing a packet from  $i$  to  $j$ . At node  $i$ , a backward ant coming from neighbor  $j$  updates the pheromone tables using the following rules (similar to those of the ABC algorithm mentioned in Section 2):

$$\tau_{ik} = \begin{cases} \frac{\tau_{ik} + \Delta\tau}{1 + \Delta\tau} & \text{if } k = j, \\ \frac{\tau_{ik}}{1 + \Delta\tau} & \text{otherwise,} \end{cases} \quad (19)$$

where  $\Delta\tau = \exp(-\gamma D_{id}^j)$ ,  $\gamma \in [0, 1]$ , and  $D_{id}^j$  is the Euclidean distance between current node  $i$  and the destination  $d$  through next hop  $j$ . The probability is higher for the nodes which are geometrically closer to the destination.

In phase two, data packets are stochastically forwarded using *data ants* (ants carrying data) relying on the routing tables built in the previous phase. If the paths towards the destination of two or more source nodes cross (or join) at some node  $k$  (i.e., they are not fully disjoint), it is very likely that the data ants associated to the different sources will experience collision at node  $k$ . The algorithm makes use of a relatively simple mechanism to arbitrate the access to the wireless channel in  $k$ : the lowest ID nodes are allowed to proceed with their transmissions first, while nodes with higher ID execute a binary exponential back-off algorithm waiting for their turn.

Ghasemaghaei et al. show through simulations that the congestion awareness of MO-IAR helps in reducing the number of collisions at the MAC layer. The performance of MO-IAR is compared with that of the SC, FF and FP algorithms proposed in [89] (see Section 7.1). The reported simulation results on randomly generated networks of 49 nodes show that MO-IAR has smaller average latency and experiences less number of collisions than its competitors. More extensive experiments are however necessary to assess the performance of the algorithm. Moreover, the congestion awareness mechanism assumes that the source nodes know about the best paths that their neighbors have discovered. However, it is not clear how this information is exchanged among neighbors. The use of source routing and the inclusion of other state information in the header of forward ants can also be problematic as already discussed earlier.

### 7.8. AntChain

AntChain, proposed by Ding and Xiaoping Liu [21], is a centralized algorithm which partitions the responsibilities of sensor nodes and the sink node according to their hardware resources and relative distances with the aim of optimizing energy consumption and transmission delays. AntChain targets the applications in which the location and the identity of the sensor nodes are known in advance (e.g., in some health care applications). The sink node exploits location information to calculate a near-optimal chain organization for the nodes, which is then used for efficient data transmission.

AntChain assumes that each sensor node can directly reach every other node in the network and can directly communicate with the sink. A node can be in one of the four states: sleeping, idle, receiving, or transmitting. In the setup phase, the sink node broadcasts a setup signal to sensor nodes, which in response send their location and ID. Using the aggregate information about locations and IDs of the operational sensor nodes, the sink node applies the Max-Min Ant System (MMAS) algorithm [84] to solve a TSP-like optimization problem to identify a near-optimal chain arrangement for the communications in WSN. Chain information is then broadcast to the sensor nodes. The node assigned to the farthest end of the chain starts a data collection and sends it to its upstream neighbor. Each intermediate node performs data aggregation and forwards the results to the next node in the chain. The process continues until the data reaches the tail node of the chain, which communicates the final result to the sink node. The nodes which are not part of the chain can go to a sleep mode until the next setup message. Repeating the setup process serves to continually refresh the chain arrangement in order to overcome problems resulting from possible node failures or battery depletion. If a node does not receive data from its downstream neighbor, it assumes that the node is not operational, and, therefore, it transmits directly its data to the base station.

AntChain's performance has been evaluated in simulation using the NS-2 simulator. The LEACH [39] and PEGASIS [54] algorithms were used for comparisons (see Section 2.2). Both these algorithms share some similarities with AntChain. Simulation results show that AntChain is more energy efficient and can provide a longer network lifetime. An important feature of this protocol is that it distinguishes between nodes with different resources. This not only reduces the processing overhead of the sensor nodes but also results in near-optimal chains by running MMAS at the sink node. Allowing nodes to

go to sleep mode saves a significant amount of energy. The main flaw of the algorithms lies in the fact of being centralized, therefore missing the robustness of fully distributed algorithms. Moreover, the assumption that each node can directly communicate with the sink node is usually hard to implement in practice.

7.9. Jumping Ant Routing Algorithm (JARA)

JARA, Chen et al. [14], combines the core characteristics of the proactive *Ant-based Routing Algorithm for MANETs* (ARAMA) [43] and of the hybrid *Zone Routing Protocol* (ZRP) [35] for MANETs. JARA divides the network into zones. Inside each zone, the nodes make use of a proactive strategy to discover routes to destinations located within the zone radius. On the other hand, a reactive ACO-based mechanism is used to discover routes to nodes located beyond the zone boundaries.

JARA assumes that each node has its own zone, and that their routes to other nodes located within the zone are known through a proactive strategy. Therefore, the algorithm only focuses on inter-zone route discovery and management, using on an ACO strategy. The nodes in each zone of radius  $\rho$  hops are categorized as interior nodes and boundary nodes (see Fig. 2). A node that can be reached in less than  $\rho$  hops from the central node is an interior node. Boundary nodes are at a distance of precisely  $\rho$  hops from the central node. When a node needs to send data to a destination located outside the zone radius, it creates multiple forward ants and each one of them is randomly sent to one of the boundary nodes. Since routes inside the zone are assumed to be known, the forward ants are seen as moving in jumps of  $\rho$  hops through the interior nodes. Next hop selection is made in a probabilistic way according to the following selection probabilities, which are of the form of Eq. (2) (but the precise form of  $f$  is left unspecified in the paper):

$$p_{ij} = \begin{cases} \frac{f(\tau_{ij}, \eta_{ij})}{\sum_{k \in N(i)} f(\tau_{ik}, \eta_{ik})} & \text{if } k = j, \\ 0 & \text{otherwise,} \end{cases} \tag{20}$$

where  $\eta_{ij}$  denotes the local heuristic value of the link  $(i, j)$  and  $\tau_{ij}$  is the pheromone value on the same link.

Backward ants use source routing to travel back to the source node and update pheromone tables using Eq. (4). Data packets are routed to their destination nodes using so-called *guide ants* that act as data carriers. The main purpose of a guide ant is to route data packets along the best found paths. The guide ant is also used when a path to a node, interior or boundary node, is broken. For instance, let us assume that node  $S$  in Fig. 2 loses connection to node  $A$ . As  $S$  is aware of its local topology, it will select an alternate node (node  $E$ ) as a relay node to reach node  $H$ , and it will launch a guide ant to reinforce the pheromone on the new path. In this way, JARA can repair broken links without initiating another route discovery mechanism.

The simulation results reported in the paper show that JARA discovers routes more quickly than ARAMA at a much less overhead. It is hard to judge this protocol since the description given in the paper is rather vague and unclear, and important concepts, like the proactive route management, are missing. Moreover, as mentioned before, WSNs have unique traffic patterns that make them quite different from generic MANETs, and probably inter-zone routing may not be fully viable in WSNs scenarios.

7.10. Energy-Delay ant-based (E-D ANTS)

Chen et al. [88] have proposed the *E-D ANTS* algorithm, that aims to find a route with minimum energy-delay product in order to maximize network lifetime and to provide a real-time data delivery service. *E-D ANTS* is a reactive protocol based on

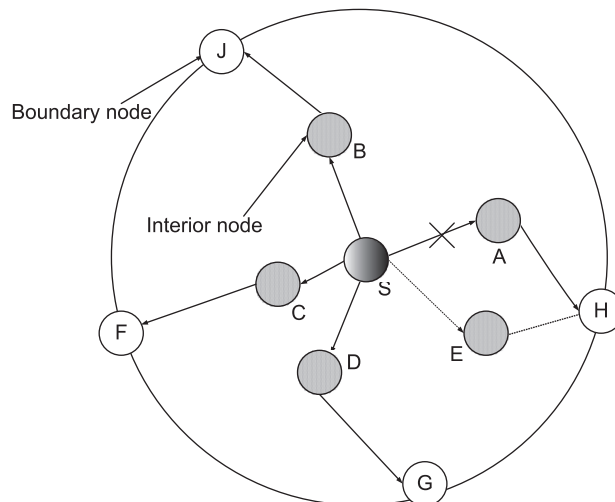


Fig. 2. JARA: nodes A, B, C, D and E are interior nodes, E, F, G and H are boundary nodes, S is the central node.

the iterative generation and unicast transmission of multiple forward ants to discover minimum energy and delay paths. The algorithm is very similar to *AntNet*. Each forward ant stores in its local memory the residual energy level and the hop delay experienced at each node. At node  $i$ , next hop  $j \in \mathcal{N}(i)$  is selected according to the following rule, analogous to the one used in *AntNet* (Eq. (1)):

$$p_{ij} = \frac{\omega\tau_{ij} + (1 - \omega)\eta_{ij}}{\omega + (1 - \omega)(|N_i| - 1)}, \quad (21)$$

where  $\omega \in [0, 1]$  weighs the relative importance of pheromone vs. heuristic values, and  $\eta_{ij}$  is given by:

$$\eta_{ij} = \frac{e_j}{\sum_{k \in \mathcal{N}(i)} e_k} \quad (22)$$

with  $e_j$  being the residual energy level of node  $j$ . The next hop is therefore probabilistically selected as the one with the best trade-off between expected latency and energy product (encoded by pheromone variables) and larger residual energy. Pheromone is updated using the following equation:

$$\tau_{ij} = \rho\tau_{ij} + \Delta\tau_{ij}. \quad (23)$$

$\Delta\tau_{ij}$  is calculated considering the relative goodness of the path sampled by the ant vs. some statistic of the goodness of the paths that have been sampled in the near past. More specifically, for each destination  $d$ , a function  $g$  is defined at each node  $i$  as the following exponential average:

$$g(k) = (1 - \gamma)g(k - 1) + \gamma e_{ij}^\alpha (t_{jd}^i)^\beta, \quad (24)$$

where  $k$  is a discrete index which is incremented after the arrival of a backward ant,  $e_{ij}$  is the energy cost of using the link  $(i, j)$ ,  $t_{jd}^i$  is the latency experienced by the ant going from  $i$  to  $d$  through  $j$ ,  $\gamma \in [0, 1]$  is a learning rate, and  $\alpha$  and  $\beta$  are positive weighting constants. If the most recent ants have followed low cost paths in terms of the weighted product between energy and latency, then  $g$  will decrease, otherwise it will increase.  $g$  defines an adaptive comparison baseline to evaluate ant paths over time (this way of proceeding is very similar to *AntNet's* behavior). If  $z_{new}$  is the energy  $\times$  latency cost of the path sampled by the  $k$ th backward ant arriving at node  $i$  from destination  $d$ , then  $\Delta\tau_{ij}$  is calculated as:

$$\Delta\tau_{ij} = \tau_0 \left( 1 - \frac{z_{new} - g(k)}{\bar{z} - g(k)} \right), \quad (25)$$

where  $\bar{z}$  is the average goodness of the last  $n$  sampled paths, and  $n$  affects algorithm responsiveness.

The authors of *E-D ANTS* used the OPNET simulator for the empirical evaluation of their algorithm and have compared its performance to that of *AntNet* and *AntChain* (see Section 7.8). The results reported in the paper show that *E-D ANTS* converges faster than the other two algorithms to near-optimal paths, at the expense of a lower routing overhead. The interesting feature of *E-D ANTS* is that it routes packets through shorter and energy-abundant paths, and also avoids congested paths. Being flat in nature, *E-D ANTS* can however hardly scale to large topologies without the inclusion of specific hierarchical mechanisms.

### 7.11. Probabilistic, Zonal and Swarm-inspired system for Wildfire Detection (PZSWiD)

Ramachandran et al. [66] have proposed PZSWiD which is a cluster-based ACO-inspired system for wildfire detection. In PZSWiD, nodes perform two functions: (i) respond to different queries that are generated by a sink node, and (ii) transport detected events (e.g., the outbreak of a fire) to the sink node. It can work with both event- and query-based applications (a feature absent in most of the other protocols). The algorithm routes queries of a sink node to the zone where it can be answered with a high probability. The sensor nodes can also generate periodic reports or emergency reports depending on the criticality of sensed data and then transport them to the sink node in a proactive manner. An example of a regular report is the periodic communication of the ambient temperature. On the other hand, an emergency report might be triggered when the temperature exceeds a certain threshold value indicating the outbreak of a fire.

The protocol is complex and the description of its different components is rather vague. The algorithm assigns a probability  $p_{it}$  of satisfying a query sent from sink  $i$  to each node  $t$  in the network. Each query consists of  $n$  records. The probabilities  $p_{it}$  are assigned on the basis of two factors: (1) how closely locally sensed data matches with the queried data, (2) the amount of pheromone. The  $p_{it}$  values are dynamically updated by the moving ants. As a result, sink queries are routed to the nodes that have higher probability of occurrence of data.  $p_{it}$  at node  $t$  is defined as:

$$p_{it} = \frac{\sum_{r=1}^n C_{R_r=R_d}}{C_n} P_t, \quad (26)$$

where  $C$  represents the number of matches found,  $R_i$  represents an individual record of the query,  $R_d$  is the data sensed by node  $t$  and  $P_t$  is the total pheromone strength at  $t$ . Ants in PZSWiD are simply the data packets (i.e., reports or responses to the sink queries) that are used to update the pheromone on their way to the sink node. Pheromone is updated using the following rule:

$$\tau_{ik} = \begin{cases} \frac{T_e F_e (1-\rho) \tau_{ik} + \rho Q}{F_t \cdot TotalCost} & \text{if } k = j, \\ \frac{T_e F_e (1-\rho) \tau_{ik}}{F_t \cdot TotalCost} & \text{otherwise,} \end{cases} \quad (27)$$

where  $F_e$  is the number of factors currently affecting ants movement,  $F_t$  is the total number of factors that can affect ants movement,  $\rho$  and  $Q$  are ACO parameters and  $TotalCost$  is the total overhead including memory, communication, and processing. Paths with high pheromone values not only have high probability of answering a query, but also route the query on a low cost path. Consequently, the good paths get reinforced, while suboptimal paths are removed due to the evaporation of pheromone.

As discussed earlier, *PZSWiD* can also generate regular or emergency reports without any explicit notification from the sink. Mathematically, the threshold value  $T_e$  is computed using the probability of occurrence and the pheromone strength:

$$T_e = \begin{cases} \frac{P_t}{P_{it}} & \text{if threshold is calculated by ant } t, \\ \frac{P_t}{P_{it}} C_e & \text{if threshold is calculated by node } t, \end{cases} \quad (28)$$

where  $C_e$  represents the number of nodes that sense the temperature above a given threshold (i.e., the nodes that match a query). Intuitively speaking, it represents the speed at which fire is spreading.

The algorithm has been implemented in NS-2, considering as metrics the average energy dissipated (Joules/node/event) and the average delay considering different zone radii. The simulation results show that *PZSWiD*'s energy consumption is minimal for a zone of radius 3. The performance of the algorithm has not been compared to that of other existing protocols.

### 7.12. Ant-aggregation

Misra and Mandal [58] used the ACO metaheuristic to propose *Ant-aggregation*, a data aggregation mechanism for WSNs. This work is based on the argument that a multihop communication model coupled with in-network aggregation can significantly enhance network lifetime because of reduced energy requirements. Therefore, Misra and Mandal address the problem of optimal aggregation in a multicast tree, which is an NP-hard problem [5]. *Ant-aggregation* is based on ACO to build minimum cost aggregation trees. Forward ants either look for the shortest path to the destination or for a close by aggregation point. At node  $i$  a forward ant is unicast to the next hop  $j$  with probability:

$$p_{ij} = \left( \frac{\tau_{ij}^\alpha / \eta_{ij}^\beta}{\sum_{k \in \mathcal{N}(i)} \tau_{ik}^\alpha / \eta_{ik}^\beta} \right), \quad (29)$$

where the pheromone encodes the preference for shortest paths and the heuristic values  $\eta$  represent the node potential. A node with a low potential value means that the node is near a sink or an aggregation point.  $\alpha$  and  $\beta$  are in  $[0, 1]$  and weight the relative importance of pheromone vs. potential. In practice, node potentials  $\eta_{ij}$  are distance estimates of a nearby aggregation point or of a shortest path to destination. An aggregation point is a node where two or more ants join their paths. A forward ant terminates its journey either at the sink or when it arrives at an aggregation point (notice that this algorithm joins ants' sub-trees similarly to the DCR algorithm discussed in Section 7.6). Backward ants update both pheromone and potentials. These latter are updated as follows:

$$\eta_{ij} = \gamma C_{ij\_agg\_point} + \eta C_{ij\_dest} + k C_{ij\_corr}, \quad (30)$$

where  $C_{ij\_agg\_point}$  and  $C_{ij\_dest}$  are the estimated distances to reach, respectively, a nearby aggregation point or a destination node using  $j$  as next hop, while  $C_{ij\_corr} \in [0, 1]$  is a coefficient measuring data correlation between  $i$  and  $j$  (e.g., if  $i$  always uses  $j$  to forward its data, the two nodes are statistically correlated). A low value of  $C_{ij\_corr}$  indicates that the data of nodes  $i$  and  $j$  are highly correlated. The pheromone updating rule is the same as in Eq. (4), with the pheromone reinforcement defined as  $Q/H$ , where  $Q$  is a positive constant, and  $H$  is the distance to the destination as measured by the forward ant. The algorithm iterates with different permutations of source nodes to converge to near-optimal aggregation points. Once the algorithm converges, sources will be directing their detected events either to an aggregation point or to a sink node, whichever is closer to the source.

In a few simulation experiments, with networks of 20 up to 40 nodes, the algorithm is compared to both a greedy and an opportunistic aggregation algorithm, showing a significant gain in energy reduction.

Liao et al. [53] have extended the work of Misra and Mandal by arguing that the probability of locating an aggregation point in the *Ant-aggregation* algorithm is low. To mitigate this potential problem, they have introduced the concept of path extension, which enables multiple routes to intersect with each other in order to increase this probability. In this scheme, a sink node first floods a packet in the network, so that sensor nodes can estimate their hop distance to the sink node. Then, a node  $s$  launches an ant packet that carries the data packet to one of its neighbors (say  $j$ ) which is selected probabilistically using Eq. (2), with  $\alpha = 1$  and  $\eta_{ij}$  equal the inverse of the number of hops to reach the sink node passing by  $j$ . This ant is further broadcast by subsequent nodes until its TTL expires. The intermediate nodes generate a backward ant which is routed to the source along the reverse path. When a node  $i$  receives this ant, it updates its pheromone table by using the following equation:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \rho\Delta\tau_{ij}, \tag{31}$$

where  $\Delta\tau_{ij} = [1 + (h_i - h_j)]\Delta\omega_j$  and  $\Delta\omega_j$  is the number of hops between the source and the sink through node  $j$ ,  $h_i$  and  $h_j$  are the number of hops separating the sink node respectively from  $i$  and  $j$ . The purpose of the backward ant is to enforce the route leading to a node closer to the sink node. An additional advantage of this approach is that different paths might have overlapping nodes. As a result, packets are routed towards the overlapping nodes or aggregation points where they can be combined with other data packets to reduce the in-network traffic. The reported simulation results show that the modified approach consumes significantly lower energy than *Directed diffusion*, which performs opportunistic aggregation.

7.13. *BeeSensor*

Saleem and Farooq [71] have proposed *BeeSensor*, a bee-inspired, reactive and event-driven multipath routing protocol for WSNs. *BeeSensor* aims at energy efficiency, scalability, and long network lifetime. Energy efficiency is achieved by limiting the number of control messages, as well as of data packets through in-network aggregation. Paths are prioritized on the basis of their remaining energy levels to extend the network lifetime.

In addition to forward and backward scout agents, *BeeSensor* makes use of additional agents such as *packers*, *foragers* and *swarms*. Packers receive data packets from the upper layers of the node architecture, and hand them over to a forager for transportation to a sink node. In turn, swarms transport a group of foragers back from the sink to the source node. Foragers are the main agents that transport events from the source to a sink node. Forward scouts carry the data, and are therefore launched on reactive basis. Intermediate nodes at  $H_l$  hops (or less) away from the source, deterministically broadcast them. On the other hand, farther nodes broadcast them with a certain probability  $p_b$ . When an intermediate node  $i$  receives a forward scout from node  $j$ , it computes a reward value  $r_{is}$  according to the following equation:

$$r_{is} = \frac{E_{is}}{H_{is}}, \tag{32}$$

where  $E_{is}$  is the minimum remaining energy and  $H_{is}$  is the hop length of the path leading from node  $i$  to node  $s$ . The main idea of computing  $r_{is}$  is to prefer least hop paths. If the hop length is the same, then high energy paths are preferred. Node  $i$  stores in its cache  $r_{is}$  and the IDs of the source node, of the scout, and of the previous hop (node  $j$ ) (see Fig. 3). Reward values may be updated by future forward scouts in the following manner. If later on a node  $i$  receives a replica of a forward scout, it computes the reward value and, if the new reward value is greater than the one present in the scout cache, it updates the reward and the previous hop entries in the cache. In this way, nodes keep the reverse link entry with the highest reward value, which is then used by a backward scout to discover a path leading from the sink to the source node and vice versa.

In response to each forward scout received at the sink, a backward scout with a unique path ID  $pid$  is sent back to the source node. When a node  $i$  receives a backward scout from node  $j$ , it checks its scout cache. If matching information is found, node  $i$  updates its forwarding table (see Fig. 3) using the following information: path ID, next hop (node  $j$ ), and previous hop (available in the scout cache). Node  $i$  then forwards the backward scout to its previous hop. The process continues until the backward scout reaches the source node, where the dance number  $DN$  is calculated as:

$$DN_n = \left\lceil \frac{|\beta - (E_{max} - E_{pid}^r)|}{\gamma} \times \alpha(e) \right\rceil, \tag{33}$$

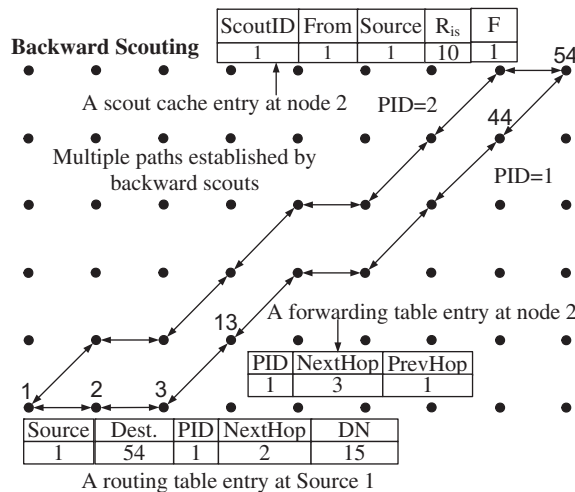


Fig. 3. *BeeSensor*: Illustration of forwarding table, routing table, and scout cache entries.

where  $E_{pid}^r$  is the minimum remaining energy of the path (identified by unique ID i.e.,  $pid$ ) reported by the backward scout,  $E_{max}$  is the initial (maximum) energy level,  $\alpha(e)$  is a function of the number of events waiting in cache and  $\beta, \gamma$  are user-defined constants. A source node then updates its routing table in which each entry contains: source node ID, destination ID, path ID, next hop and  $DN_n$ . The probability  $p_{nd}$  of selecting neighbor  $n$  as a next hop to reach sink node  $d$ , is computed as:

$$p_{nd} = \frac{DN_n}{\sum_{j=1}^l DN_j}, \quad (34)$$

where  $DN_n$  is the dance number associated with neighbor  $n$ , and  $l$  is the number of reachable neighbors. Once a route is discovered, foragers transport events to the sink. Since a forager follows a predetermined path, intermediate nodes do not need to issue routing decisions. This is different from typical ACO/ant-based routing protocols, in which intermediate nodes stochastically select the next hop. In *BeeSensor*, foragers are stochastically routed at the source, while intermediate nodes perform a deterministic forwarding based on path ID. For instance, in Fig. 3, source node 1 can select any of the two available next hop nodes. Once the forager is forwarded to one of these nodes, say node 2, it is bound to be forwarded to node 3.

*BeeSensor* does not allow neighboring nodes to initiate scouting in parallel. If a node has already started scouting, on the detection of an event, its neighbors wait for that node to announce its route. Once the first source node discovers a route, it announces this route through the broadcast of a beacon message to all its neighbors. They will then forward their detected events to this source node, that will perform in-network aggregation before forwarding the results to the sink node. There is no explicit path maintenance procedure in *BeeSensor*. The paths are intact as long as foragers are available in the routing tables. When a source node runs out of foragers, paths automatically expire.

The authors of *BeeSensor* have used the probabilistic wireless network simulator *Prowler* [81] for the empirical evaluation of the algorithm, and have compared its performance to *AODV* [63], *EEABR* (Section 7.2) and *FP* (Section 7.1.3). In the reported experiments, compared to the considered other algorithms, *BeeSensor* shows a significantly higher packet delivery ratio, a lower control overhead, and a longer network lifetime. Its packet delivery ratio is on the other hand comparable to that of the best protocol, *FP*, in terms of reliability. Saleem et al. [73] have also proposed a mathematical modeling framework for the study of the behavior and the performance of the algorithm. They have extended their evaluation framework in [72,74] which can not only be adapted to a variety of ad hoc routing protocols but is particularly useful when dealing with large scale networks.

#### 7.14. Other protocols

Bashyal et al. [7] have proposed the *Collaborative Routing Algorithm for Wireless Sensor Networks (CRAWL)*. Assuming that the nodes have a random non-uniform distribution of residual energy levels, *CRAWL* forms clusters by assigning the role of cluster heads to the nodes with larger residual energy. Consequently, *CRAWL* is quite scalable to large network topologies and can easily adapt to random distributions of residual energy levels. Bashyal et al. also suggest that the network lifetime should be a measure of how long the network performed satisfactorily. Based on this argument, they define an effectiveness metric  $Eff = \sqrt{\frac{\text{AreaCovered} \times \text{SurvivingNodes}}{\text{TotalArea} \times \text{TotalNodes}}}$ .  $Eff$  gives a quantitative measure of the effectiveness of a network after a certain percentage of nodes are dead. The reported simulation results show that *CRAWL* is able to achieve 20% better network lifetime than a non-collaborative routing algorithm.

Selvakennedy et al. [78] have proposed *T-ANT* which is a distributed, cluster-based data gathering protocol for WSNs. The major objective of *T-ANT* is to optimize network lifetime by forming evenly distributed clusters at minimal energy cost. The protocol exploits the *separation* and *alignment* principles found in biological swarms, thereby making use of a very limited number of ants to form clusters, hence incurring in limited energy overhead. The simulation results reported in the paper show that *T-ANT* attains even distribution of cluster heads and better network lifetime compared to competitor protocols.

The *Pheromone based Energy Aware Directed Diffusion PEADD* algorithm, Zhu [90], is a variant of the *Directed diffusion* protocol [44]. *PEADD* is based on ACO. The major objective of the algorithm is to extend the network lifetime by involving high energy nodes in the data gathering process. *PEADD*'s ants reinforce the pheromone on a path proportionally to the remaining energy levels of the nodes. Paths with larger residual energy are positively reinforced, while the others are negatively reinforced. The simulation results reported in the paper show that for an increasing number of dead nodes, the network lifetime achieved by *PEADD* is significantly higher compared to that achieved by *Directed diffusion*.

Paone et al. [61] have proposed an SI-based routing algorithm which is designed to be fault-tolerant, self-organized, and adaptive to the changing environment. The authors of the algorithm define  $FA_i$ , the forwarding attitude of a node  $i$  as a function of its battery charge  $b$ ,  $FA_i : b \in \mathbb{R} \rightarrow ]0, 1[$ .  $FA_i$  is used to reinforce pheromone values. The algorithm has a proactive nature. The sink node broadcasts its  $FA$  to its neighbors, which in turn calculate their own  $FA$  using  $FA = FA_e \cdot FA_i$ , where  $FA_e = \frac{\sum_{j=1}^n \tau^j}{n}$ , with  $n$  being the number of neighbors, and  $\tau^i$  is the perceived pheromone level. Also the sensor nodes periodically exchange their  $FA$  values, and, as a result, they form a pheromone gradient towards the sink. Simulation results show that, varying the characteristics and the size of the network, the packet delivery ratio of the proposed algorithm remains constant, showing the good scalability of the algorithm.

Chen et al. [13] have introduced an improved ant-based routing algorithm that seems to converge quicker than other ant-based routing algorithms. The algorithm makes use of search ants, which are generated by the sink node and guide the subsequent forward ants so that they can quickly reach the sink node. The Euclidean distance is used as the measure of the cost

of a link and for assigning pheromone values. The technique used for guiding forward ants is similar to the one used in [12] and [89].

There are a few other variants of ant-based algorithms for WSNs reported in literature. The interested reader can consult, for instance, the following additional references: [6,41,42,52,56,60,65].

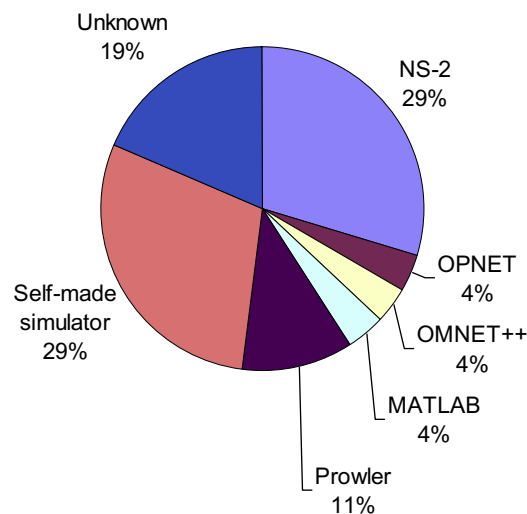
## 8. A critical discussion on the scientific soundness of the reviewed work on WSN routing

From the number of papers that we have reviewed it is clear that, so far, significant efforts have been addressed to exploit SI techniques to design effective routing protocols for WSNs. In this section we report and discuss some statistics related to the way these protocols have been presented and evaluated, and we build on these results to provide some indications about future steps that should be taken in the field in order to reach true state-of-the-art quality.

The results from our statistical elaborations are reported in Table 2 and in Fig. 4. In general terms, these statistics point out three major shortcomings common to the large majority of the considered works: (i) the simulation environment is not satisfactorily described, (ii) algorithms are not evaluated over a large set of operational scenarios, (iii) most of the protocols are not compared with other state-of-the-art protocols for routing in WSNs. Therefore, if from the one hand the reported protocols seem to have appealing properties and good performance, from the other hand, the way they are actually presented and evaluated seems to lack true scientific soundness. The critical discussion that follows precisely aims at pointing out the most evident deficiencies from the methodological point of view, and, at the same time, at promoting new ways to present and validate the promising features of SI-based design for WSN routing.

**Table 2**  
Statistics about the methodological approach followed in the reviewed papers.

Description	Percentage papers
Perform empirical evaluation of algorithms	100
Compare to other algorithms	52
Make a public version of protocol implementation	0
Do not specify simulator name or use a self-made one	47
Report the number of nodes in the network	93
Report simulation length	44
Report the transmission radius of the nodes	40
Report the number of iterations	37
Report the size of simulation area	74
Report packet delivery ratio	26
Report energy consumed or a related metric	48
Report average packet (data) latency	33
Report routing overhead	15
Report network lifetime	15



**Fig. 4.** Simulator usage for the empirical evaluation of the algorithms.

### 8.1. Used simulation tools and description of the simulation

All the reviewed protocols have been evaluated in simulation. The choice of a realistic simulation tool is of fundamental importance to be able to run meaningful experiments and to carry out a sound empirical validation of the developed algorithms. This is especially true in the context of wireless networks (e.g., see [16] for a general discussion). In Fig. 4, we report a pie chart representation of the distribution of the different simulators used in the reviewed papers. We can see that a number of different simulators have been used, with the well-known NS-2 simulator and custom-made simulators being the most popular choices, followed by a good percentage of cases in which the used simulator is not even mentioned (19%). This is a quite disappointing situation. Actually most authors have provided little or no information about the used simulation environment.

The use of self-made simulators is the most questionable choice in this context. In fact, first, it does not allow, in general, the repeatability of the experiments. Second, as a matter of fact, the proper simulation of a complex wireless environment is a hard task that usually requires the collaboration of many experts in the field and the refinement and the debugging of the software over relatively long time periods. Therefore, it is very hard to expect a self-made simulator being satisfactorily realistic.

Also the use of simulators that have not been specifically designed for WSNs but that are just adapted to them can be questionable. This is for instance also the case of NS-2, which is more suitable for wired networks and MANETs. An up-to-date overview of existing simulators for WSNs can be found in [50].

Another negative aspect that emerges from the reviewed literature is that also the way the simulator has been set up and used is often not clearly explained. For instance, only 44.5% of the papers report the total duration of the simulation and only 40% report the transmission range of a node, which is a basic type of information. About 26% of the papers do not even describe the size of the simulation area. We did not investigate in detail all the different aspects characterizing a simulation, but, for instance a very few papers report information about the depletion model for the nodes or about the protocols used at the MAC layer.

### 8.2. Experimental evaluation methodology

A sound evaluation and assessment of the characteristics and performance of a routing protocol is one of the most critical and difficult steps in the protocol engineering cycle. On the other hand, in the case of the large majority of the reviewed protocols:

1. Little or no information at all is provided to describe the tools used for simulation and the setup of the experiments, that also hinders repeatability, as already pointed out.
2. Statistical significance of the reported data and of the carried experiments looks quite weak. For instance, only 37% of the authors report the number of trials used to take the average value of the selected performance metrics, and no variance is actually reported regarding these values. In the case of performance comparison with other algorithms, no statistical hypothesis tests are carried out to assess the significance of the relative difference in performance.
3. Usually results are presented with no discussion. On the other hand, the reasons behind the good or bad performance of a protocol must be properly justified and explained to be able to learn what the effective and the ineffective components of an algorithm are.
4. In many cases, no comparisons with other state-of-the-art algorithms are carried out, which is on the other hand, a necessary step given the empirical nature of the addressed domain.
5. Only a few, often small-sized, scenarios are considered in the experiments. Moreover, some authors report just a single metric (e.g., cost of a path in terms of hops or energy), in a single scenario. Some authors do not even define their performance metrics properly.

As the result of these incorrect methodologies, the evaluation procedure that has been followed in the large majority of the considered algorithms looks biased, unrepeatable, and unreliable. In order to promote the field of SI-based routing in WSNs, which we believe is a very promising application area given the peculiar characteristics of SI algorithms, in the following we outline some essential methodological aspects that should be taken into account in future works.

- A proper network simulator must be selected, possibly one that has been developed specifically for WSNs. The details of the simulation settings (e.g., number of nodes, communication radius, MAC layer, deployment area, energy depletion model, simulation time) must be clearly stated.
- Performance metrics must be clearly defined. Some basic performance measures must always be taken into account, such as average end-to-end delay, packet delivery ratio, energy efficiency (energy per packet or energy per bit of delivered data), and routing overhead. In addition, protocol-specific metrics (e.g., optimality of clustering topology or of the built multicast trees, network lifetime) must also be included in performance evaluation.
- The comparison with different state-of-the-art algorithms for WSNs, possibly selected from the networking literature, is customary. The selected algorithms must have been designed for WSNs, and not for wired networks (e.g., *AntNet*) or for MANETs (e.g., AODV). Appropriate statistical tests (both parametric or non-parametric) must be used in order to assess the relative difference in performance in a way which is statistically sound.

- The algorithm must be evaluated in terms of robustness to losses and noise, in terms of adaptivity to changes in data traffic and topology, and in terms of scalability. The latter is particularly important for WSNs. Therefore a number of different network scenarios must be considered, ranging from small to rather large networks.
- In order to favor fair comparisons among different algorithms, and at the same time promote their use, the authors should make their code available on the Internet.
- When developing an algorithm, the designer should always keep in mind the associated computational requirements. WSNs are made of resource-constrained nodes. Therefore, the algorithms should be less demanding as possible from a computational point of view.
- Algorithm description should be clear and compact, and always accompanied by a pseudo-code.

## 9. Conclusions and future directions

Wireless sensor networks consist of large sets of resource-constrained nodes. The design of effective, robust, and scalable routing protocols in these networks is a challenging task. On the other hand, the relatively novel domain of swarm intelligence offers algorithmic design principles, inspired by complex adaptive biological systems, that well match the constraints and the challenges of WSNs. Therefore, a number of routing protocols for WSNs have been developed in the last years based on SI principles, and, more specifically, taking inspiration from foraging behaviors of ant and bee colonies.

In this paper, we have presented a rather extensive survey of these SI-based algorithms for routing in WSNs. We have also pointed out a number of methodological flaws in the way these algorithms are commonly presented and empirically evaluated. Finally, we have outlined a general recipe for the definition of scientifically sound experiments and performance evaluation.

To conclude, we want to sketch some future directions for the field. Apart from the highlighted methodological problems, the domain of SI-based WSN routing lacks of contributions falling in the two opposite areas of mathematical modeling and real-world implementations. From the one hand, simulation-based studies should be complemented with mathematical models, that allow to study very large systems and general algorithm properties, and can favor fair comparisons among the algorithms. From the other hand, simulation should be just the first step towards hardware implementations. Experiments with real testbeds force the experimenter to face a wide set of problems and challenges that can hardly be replicated in simulation.

We strongly believe that we will witness a large diffusion of SI-based solutions for real-world WSNs once SI researchers will commit themselves to the use of sound experimental methodologies, and will carry out studies integrating mathematical modeling, simulation, and real-world testing.

## References

- [1] A.A. Abbasi, M. Younis, A survey on clustering algorithms for wireless sensor networks, *ACM Computer Communications* 30 (14–15) (2007) 2826–2841.
- [2] K. Akkaya, M. Younis, A survey on routing protocols for wireless sensor networks, *Ad Hoc Networks* 3 (3) (2005) 325–349.
- [3] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, *IEEE Communications Magazine* 40 (8) (2002) 102–114.
- [4] J. Al-Karaki, A. Kamal, Routing techniques in wireless sensor networks: a survey, *IEEE Wireless Communications* 11 (6) (2004) 6–28.
- [5] J. Al-Karaki, R. Ul-Mustafa, A. Kamal, Data aggregation in wireless sensor networks – exact and approximate algorithms, in: *Proceedings of the International Workshop on High-Performance Switching and Routing*, 2004.
- [6] F. Anoosha, R. Shokri, N. Yazdani, A. Nayyeri, Antmig: a novel code migration method to conserve energy in wireless sensor networks, in: *Proceedings of the IEEE Wireless Communications & Networking Conference (WCNC)*, 2008.
- [7] S. Bashyal, G. Kumar, Venayagamoorthy, Collaborative routing algorithm for wireless sensor network longevity, in: *Proceedings of the IEEE International Conference on Intelligent Sensors, Sensor Networks and Information (ISSNIP)*, 2007.
- [8] F.V.D. Bergh, A. Engelbrecht, A study of particle swarm optimization particle trajectories, *Information Sciences* 176 (8) (2006) 937–971.
- [9] E. Bonabeau, M. Dorigo, G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, New York, USA, 1999.
- [10] A. Boukerche, M. Ahmad, B. Turgut, D. Turgut, A taxonomy of routing protocols in sensor networks, in: A. Boukerche (Ed.), *Algorithms and Protocols for Wireless Sensor Networks*, Wiley, 2008, pp. 129–160 (Chapter 6).
- [11] W. Cai, X. Jin, Y. Zhang, K. Chen, R., Wang, ACO based QoS routing algorithm for wireless sensor networks, in: *Proceedings of the 3rd International Conference on Ubiquitous Intelligence and Computing (UIC)*, LNCS, vol. 4159, 2006.
- [12] T. Camilo, C. Carreto, J.S. Silva, F. Boavida, An energy-efficient ant-based routing algorithm for wireless sensor networks, in: *Proceedings of the 5th International Workshop on Ant Colony Optimization and Swarm Intelligence (ANTS)*, LNCS, vol. 4150, Springer, Berlin, Germany, 2006, pp. 49–59.
- [13] G. Chen, T.-D. Guo, W.-G. Yang, T. Zhao, A swarm-based routing protocol for wireless sensor networks, in: *Proceedings of IEEE International Conference on Collaborative Computing: Networking, Applications and Work-sharing*, 2006.
- [14] W.-M. Chen, C.-S. Li, F.-Y. Chiang, H.-C. Chao, Jumping ant routing algorithm for sensor networks, *Computer Communications* 30 (14–15) (2007) 2892–2903.
- [15] Crossbow, *Wireless sensor networks, product reference guide*, 2007. <<http://www.xbow.com/Products/wdownloadCatalog.aspx>>.
- [16] G.A. Di Caro, *Analysis of Simulation Environments for Mobile Ad Hoc Networks*, Tech. Rep. 24-03, IDSIA, Lugano, Switzerland, 2003.
- [17] G.A. Di Caro, *Ant Colony Optimization and Its Application to Adaptive Routing in Telecommunication Networks*, Ph.D. Thesis, Faculté des Sciences Appliquées, Université Libre de Bruxelles (ULB), Brussels, Belgium, 2004.
- [18] G.A. Di Caro, M. Dorigo, AntNet: distributed stigmergetic control for communication networks, *Journal of Artificial Intelligence Research (JAIR)* 9 (1998) 317–365.
- [19] G.A. Di Caro, F. Ducatelle, L. Gambardella, AntHocNet: an adaptive nature-inspired algorithm for routing in mobile ad hoc networks, *European Transactions on Telecommunications (ETT)* 16 (2) (2005) 443–455 (Special Issue on Self Organization in Mobile Networking).
- [20] G.A. Di Caro, F. Ducatelle, L. Gambardella, Theory and practice of Ant Colony Optimization for routing in dynamic telecommunication networks, in: N. Sala, F. Orsucci (Eds.), *Reflecting Interfaces: The Complex Coevolution of Information Technology Ecosystems*, Idea Group, Hershey, PA, USA, 2008, pp. 11–32.

- [21] N. Ding, P. Xiaoping Liu, Data gathering communication in wireless sensor networks using ant colony optimization, in: Proceedings of IEEE International Conference on Robotics and Biomimetics, 2004.
- [22] M. Dorigo, G.A. Di Caro, The ant colony optimization metaheuristic, in: D. Corne, M. Dorigo (Eds.), *New Ideas in Optimization*, McGraw-Hill, 1999, pp. 11–32.
- [23] M. Dorigo, G.A. Di Caro, L. Gambardella, Ant algorithms for discrete optimization, *Artificial Life* 5 (2) (1999) 137–172.
- [24] M. Dorigo, T. Stützle (Eds.), *Ant Colony Optimization*, MIT press, 2004.
- [25] F. Ducatelle, G.A. Di Caro, L. Gambardella, Using ant agents to combine reactive and proactive strategies for routing in mobile ad hoc networks, *International Journal of Computational Intelligence and Applications* 5 (2) (2005) 169–184 (Special Issue on Nature-Inspired Approaches to Networks and Telecommunications).
- [26] F. Ducatelle, G.A. Di Caro, L. Gambardella, Principles and applications of swarm intelligence for adaptive routing in telecommunications networks. *Swarm Intelligence*, in press. doi:10.1007/s11721-010-0040-x.
- [27] A. Engelbrecht, *Computational Intelligence: An Introduction*, second ed., Wiley, 2007.
- [28] P. Eugster, P. Felber, R. Guerraoui, A.-M. Kermerrec, The many faces of publish/subscribe, *ACM Computing Surveys* 35 (2) (2003) 114–131.
- [29] M. Farooq, *Bee-Inspired Protocol Engineering: from Nature to Networks*, Natural Computing Series, Springer, 2009.
- [30] M. Farooq, G.A. Di Caro, Routing protocols inspired by insect societies, in: C. Blum, D. Merkle (Eds.), *Swarm Intelligence, Introduction and Applications*, Natural Computing Series, Springer-Verlag, 2008, pp. 101–160.
- [31] R. Ghasemaghaei, M.A. Rahman, W. Gueaieb, A. El Saddik, Ant colony-based reinforcement learning algorithm for routing in wireless sensor networks, in: Proceedings of the IEEE Instrumentation and Measurement Technology Conference (IMTC), 2007, pp. 2173–2178.
- [32] R. Ghasemaghaei, M.A. Rahman, W. Gueaieb, A. El Saddik, Ant colony-based many-to-one sensory data routing in wireless sensor networks, in: Proceedings of the IEEE International Conference on Computer Systems and Applications.
- [33] A. Ghosh, A. Halder, M. Kothari, S. Ghosh, Aggregation pheromone density based data clustering, *Information Sciences* 178 (13) (2008) 2816–2831.
- [34] S. Goss, S. Aron, J.L. Deneubourg, J.M. Pasteels, Self-organized shortcuts in the Argentine ant, *Naturwissenschaften* 76 (1989) 579–581.
- [35] Z. Haas, M. Pearlman, P. Samar, Zone routing protocol. IETF Internet Draft, 2002. <<http://people.ece.cornell.edu/haas/wnl/Publications/draft-ietf-manet-zone-zrp-04.txt>>.
- [36] Z.J. Haas, T. Small, A new networking model for biological applications of ad hoc sensor networks, *IEEE/ACM Transactions on Networking* 14 (2006) 27–40.
- [37] Z. He, B.S. Lee, X.S. Wang, Aggregation in sensor networks with a user-provided quality of service goal, *Information Sciences* 178 (9) (2008) 2128–2149.
- [38] W. Heinzelman, A. Chandrakasan, H. Balakrishnan, Energy-efficient communication protocol for wireless microsensor networks, in: Proceedings of the 33rd IEEE Hawaii International Conference on System Sciences (HICSS), 2000.
- [39] W. Heinzelman, A. Chandrakasan, H. Balakrishnan, An application-specific protocol architecture for wireless microsensor networks, *IEEE Transactions on Wireless Communications* 1 (4) (2002) 660–670.
- [40] W. Heinzelman, J. Kulik, H. Balakrishnan, Adaptive protocols for information dissemination in wireless sensor networks, in: Proceedings of the ACM/IEEE MobiCom Conference, 1999, pp. 174–185.
- [41] R. Huang, J. Zhu, X. Yu, The ant-based algorithm for the optimal many-to-one routing in sensor networks, in: Proceedings of the IEEE International Conference on Communications, Circuits and Systems, 2006.
- [42] R. Huang, J. Zhu, X.-T. Yu, The ant-based algorithm for the data gathering routing structure in sensor networks, in: Proceedings of the IEEE International Conference on Machine Learning and Cybernetics, 2006.
- [43] O. Hussein, T. Saadawi, Ant routing algorithm for mobile ad-hoc networks ARAMA, in: Proceedings of the IEEE Performance Computing and Communications Conference (IPCCC), 2003.
- [44] C. Govindan, R. Intanagonwiwat, D. Estrin, J. Heidemann, F. Silva, Directed diffusion for wireless sensor networking, *IEEE/ACM Transactions on Networking* 11 (1) (2003) 2–16.
- [45] S.S. Iyengar, H.C. Wu, N. Balakrishnan, S.Y. Chang, Biologically inspired cooperative routing for wireless mobile sensor networks, *IEEE Systems Journal* 1 (1) (2007) 29–37.
- [46] Z. Jin, Y. Jian-Ping, Z. Si-Wang, L. Ya-Ping, L. Guang, A survey on position-based routing algorithms in wireless sensor networks, *Algorithms* 2 (1) (2009) 158–182.
- [47] J. Kennedy, R.C. Eberhart, Y. Shi, *Swarm Intelligence*, Morgan Kaufman, San Francisco, USA, 2001.
- [48] J. Kephart, D. Chess, The vision of autonomic computing, *IEEE Computer Magazine* 36 (1) (2003) 41–50.
- [49] Y. Kiri, M. Sugano, M. Murata, Self-organized data-gathering scheme for multi-sink sensor networks inspired by swarm intelligence, in: Proceedings of the IEEE International Conference on Self-Adaptive and Self-organizing Systems, 2007.
- [50] M. Korkalainen, M. Sallinen, N. Kärrkäinen, P. Tukeyva, Survey of wireless sensor networks simulation tools for demanding applications, in: Proceedings of the Fifth IEEE International Conference on Networking and Services (ICNS), 2009.
- [51] B. Krishnamachari, D. Estrin, S. Wicker, Modeling data-centric routing in wireless sensor networks, in: Proceedings of the IEEE International Conference on Computer Communications (INFOCOM), 2002.
- [52] W.-H. Liao, Y. Kao, C.-M. Fan, An ant colony algorithm for data aggregation in wireless sensor networks, in: Proceedings of the IEEE SENSORCOMM Conference, 2007.
- [53] W.-H. Liao, Y. Kao, C.-M. Fan, Data aggregation in wireless sensor networks using ant colony algorithm, *Journal of Network and Computer Applications* 31 (4) (2008) 387–401.
- [54] S. Lindsey, C. Raghavendra, PEGASIS: power-efficient gathering in sensor information systems, in: Proceedings of IEEE Aerospace Conference, 2002.
- [55] J. Liu, F. Zhao, D. Petrovic, Information-directed routing in ad hoc sensor networks, *IEEE Journal on Selected Areas in Communications* 23 (2005) 851–861.
- [56] Y. Liu, H. Zhu, K. Xu, Y. Jia, A routing strategy based on ant algorithm for WSN, in: Proceedings of the 3rd IEEE International Conference on Natural Computation (ICNC), 2007.
- [57] F. Marcelloni, M. Vecchio, Enabling energy-efficient and lossy-aware data compression in wireless sensor networks by multi-objective evolutionary optimization, *Information Sciences* 180 (10) (2010) 1924–1941.
- [58] R. Misra, C. Mandal, Ant-aggregation: ant colony algorithm for optimal data aggregation in wireless sensor networks, in: Proceedings of IEEE International Conference on Wireless and Optical Communications Networks, 2006.
- [59] C. Ok, S. Lee, P. Mitra, S. Kumara, Distributed routing in wireless sensor networks using energy welfare metric, *Information Sciences* 180 (9) (2010) 1656–1670.
- [60] S. Okdem, D. Karaboga, Routing in wireless sensor networks using Ant Colony Optimization, in: Proceedings of the First NASA/ESA Conference on Adaptive Hardware and Systems (AHS), 2006.
- [61] M. Paone, L. Paladina, D. Bruneo, A. Puliafito, A swarm-based routing protocol for wireless sensor networks, in: Proceedings of the IEEE International Symposium on Network Computing and Applications, 2007.
- [62] C. Perkins, P. Bhagwat, Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers, in: Proceedings of the ACM SIGCOMM Conference on Communications Architectures, Protocols and Applications, 1994, pp. 234–244.
- [63] Perkins, C.E., Royer, E.M., 1999. Ad-hoc on-demand distance vector routing, in: Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications.
- [64] C. Raghavendra, K. Krishna, T. Znati (Eds.), *Wireless Sensor Networks*, Springer-Verlag, 2004.
- [65] M.A. Rahman, R. Ghasemaghaei, A. El Saddik, W. Gueaieb, M-IAR: biologically inspired routing protocol for wireless multimedia sensor networks, in: Proceedings of the IEEE International Instrumentation and Measurement Technology Conference (IMTC), 2008, pp. 1823–1827.

- [66] C. Ramachandran, S. Misra, M.S. Obaidat, A probabilistic zonal approach for swarm-inspired wildfire detection using sensor networks, *Wiley InterScience International Journal of Communication Systems* 21 (10) (2008) 1047–1073.
- [67] H. Ren, M.Q.-H. Meng, Biologically inspired approaches for wireless sensor networks, in: *Proceedings of IEEE the International Conference on Mechatronics and Automation*, 2006.
- [68] Q. Ren, Q. Liang, Energy and quality aware query processing in wireless sensor database systems, *Information Sciences* 177 (10) (2007) 2188–2205.
- [69] E.M. Royer, C.-K. Toh, A review of current routing protocols for ad hoc mobile wireless networks, *IEEE Personal Communications* 6 (2) (1999) 46–55.
- [70] N. Sadagopan, B. Krishnamachari, A. Helmy, Active query forwarding in sensor networks, *Ad Hoc Networks* 3 (2005) 91–113.
- [71] M. Saleem, M. Farooq, Beesensor: a bee-inspired power aware routing protocol for wireless sensor networks, in: *Proceedings of the 4th EvoCOMNET Workshop*, LNCS, vol. 4448, 2007.
- [72] M. Saleem, Israr Ullah, S.A. Khayam, M. Farooq, On the reliability of ad hoc routing protocols for loss-and-delay sensitive applications, *Ad Hoc Networks* (2010), doi:10.1016/j.adhoc.2010.07.012.
- [73] M. Saleem, S.A. Khayam, M. Farooq, A formal performance modeling framework for bio-inspired ad hoc routing protocols, In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2008.
- [74] M. Saleem, S.A. Khayam, M. Farooq, On performance modeling of ad hoc routing protocols, *EURASIP Journal on Wireless Communications and Networking (JWCN)* (2010). doi:10.1155/2010/373759.
- [75] P.S. Sausen, M.A. Spohn, A. Perkusich, Broadcast routing in wireless sensor networks with dynamic power management and multi-coverage backbones, *Information Sciences* 180 (5) (2010) 653–663.
- [76] R. Schoonderwoerd, O.E. Holland, J.L. Bruten, L.J.M. Rothkrantz, Ant-based load balancing in telecommunication networks, *Adaptive Behavior* 5 (2) (1996) 169–207.
- [77] T.D. Seeley (Ed.), *The Wisdom of the Hive*, Harvard University Press, London, 1995.
- [78] S. Selvakennedy, S. Sinnappan, Y. Shang, A biologically-inspired clustering protocol for wireless sensor networks, *Computer Communications* 30 (14–15) (2007) 2786–2801.
- [79] N. Shrivastava, C. Buragohain, D. Agrawal, S. Suri, Medians and beyond: new aggregation techniques for sensor networks, in: *Proceedings of the 2nd ACM International Conference on Embedded Networked Sensor Systems (SenSys)*, 2004.
- [80] K.M. Sim, W.H. Sun, Ant colony optimization for routing and load balancing: survey and new directions, *IEEE Transactions on System, Man and Cybernetics* 33 (5) (2003) 560–572.
- [81] G. Simon, Probabilistic wireless network simulator, 2004. <<http://www.isis.vanderbilt.edu/projects/nest/prowler/>>.
- [82] G. Singh, S. Das, S. Gosavi, S. Pujar, Ant colony optimization for Steiner trees: Application to routing sensor networks, in: L.D. Castro, F. Von Zuben (Eds.), *Recent Developments in Biologically Inspired Computing*, Idea Group, Hershey, PA, USA, 2005, pp. 181–206.
- [83] K. Sohraby, D. Minoli, T. Znati, *Wireless Sensor Networks: Technology, Protocols, and Applications*, Wiley, 2007.
- [84] T. Stützle, H. Hoos, *MA<sub>TS</sub>-M<sub>FN</sub>* ant system, *Future Generation Computer Systems* 16 (8) (2000).
- [85] Y. Sun, H. MA, L. Liu, Y. Zheng, ASAR: an ant-based service-aware routing algorithm for multimedia sensor networks, *Frontiers of Electrical and Electronic Engineering in China* 3 (1) (2008) 25–33.
- [86] D. Teodorovic, Bee colony optimization (bco), in: C.P. Lim, L.C. Jain, S. Dehuri (Eds.), *Innovations in Swarm Intelligence*, Springer-Verlag, 2009, pp. 39–60.
- [87] H.F. Wedde, M. Farooq, A comprehensive survey of nature-inspired routing protocols for telecommunication networks, *Journal of System Architecture* 52 (8) (2006) 461–484.
- [88] Y.-F. Wen, Y.-Q. Chen, M. Pan, Adaptive ant-based routing in wireless sensor networks using energy delay metrics, *Springer's Journal of Zhejiang University – Science A* 9 (4) (2008) 531–538.
- [89] Y. Zhang, L. Kuhn, M. Fromherz, Improvements on ant routing for sensor networks, in: *Proceedings of the 4th International Workshop on Ant Colony Optimization and Swarm Intelligence (ANTS)*, LNCS, vol. 3172.
- [90] X. Zhu, Pheromone based energy aware directed diffusion algorithm for wireless sensor network, in: *Proceedings of the International Conference on Intelligent Computing (ICIC)*, LNCS, vol. 4681, 2007.