Machine Learning Techniques Applied to Wireless Ad-Hoc Networks: Guide and Survey

Anna Förster
University of Lugano, Switzerland
Email: anna.egorova.foerster@lu.unisi.ch

Abstract— **This work is a survey on the usage of machine learning techniques in wireless sensor networks (WSNs) and mobile ad-hoc networks (MANETs). Its focus lies on approaches for data routing. The goal of the work is two-fold: first, to classify and evaluate the most important existing and on-going research in the area and, second, to provide a guide for researchers wanting to apply machine learning techniques. For this, it also gives short description of the most appropriate algorithms and suggests for which scenarios they can be best used.**

I. Introduction

Traditionally, protocols and applications in the networking domain have been designed to work in large scale heterogeneous, hierarchically organized networks with low failure rate. Addresses are assigned to nodes and the nodes keep static routing tables in order to be able to find the path to some other node. If some topology change occurs, like failing or mobile nodes, the changed routing information must be propagated through the whole network. In result, latency and loss of data increase, until the new information is successfully transmitted.

In a wireless ad-hoc scenario, new problems arise and traditional routing protocols cannot be successfully applied. Infrastructure and a well-organized hierarchy are missing, so topology changes need to propagate through the *whole* network, not only up to the next hierarchical level. In the same time, mobility and node failures increase dramatically. Transmitting of data becomes much more expensive, and the quality of service suffers. Additionally, in energy-restricted environments like WSNs the overhead of keeping routing information fresh becomes unbearable.

Given the problem context, many researchers have turned their attention to the domain of *machine learning (ML)*. The goal of this class of algorithms is to automatically *learn* the properties of the environment and to adapt their behavior quickly and easily to them. Different properties of usual WSN applications [1], [2] have to be considered: memory and computational limitations, communication costs, restricted energy. On the other hand, many ML techniques exist and their applicability to the networking domain is unclear.

This paper is both a survey and a guide: a survey for existing works and a guide for researchers willing to apply machine learning to problems in ad hoc networking. First, Section II gives short descriptions of suitable machine learning approaches for use in wireless ad-hoc networks, identifying their most relevant requirements and advantages and suggesting their applicability domain. Section III classifies and evaluates the most important existing and on-going research in the area. The presented works are evaluated in terms of their quality – meeting the application requirements. A summary of the presented work together with some future research directions in the area are presented in Section IV.

II. Finding the right algorithm

Different properties of wireless ad-hoc networks [1], [2] have to be considered when selecting a good ML approach.

Wireless ad-hoc nature: there is no fixed infrastructure nor a hierarchical structure. The shared wireless medium sets additional restrictions on the possible communication between the nodes and new problems arise, like asymmetric links.

Mobility and topology changes: MANETs are usually highly mobile scenarios, where new nodes constantly join the network, and existing ones either move through the network or leave it. Applications have to take this into account and to be able to cope with the flexible topology of the network. On the other hand, energy is not restricted, since individual nodes are usually laptops or PDAs, which can be easily recharged.

Energy limitations: On the other hand, WSNs are rarely very mobile, but highly energy-restricted. The basic scenario includes a fixed topology of sensor nodes, together with a limited number of more powerful base stations and no maintenance or recharging is possible after deployment. Therefore, cost minimization and autonomous behavior are desirable.

Physical distribution: Each node in a MANET or a WSN is an autonomous computational unit, which communicates with its neighbors via messages. Data is also distributed throughout the nodes of the network and can be gathered on a central station only with high communication costs. Consequently, algorithms requiring global information from the whole network become very expensive. Thus, distributability of the algorithms is highly recommended.

In distributed ML algorithms, the solution is found in a collaborative manner and each node holds only the part of it, directly relevant to its actions (like the best next hop in a routing problem). The remainder of this section presents a short overview of the most widely applied distributed ML techniques, together with their advantages/disadvantages and properties relevant to the wireless ad hoc scenario. A summary of their properties is given in Table I.

A. Reinforcement Learning - RL

Reinforcement learning [3], [4] is biologically inspired and acquires its knowledge by actively exploring its environment.

1-4244-1502-0/07/$25.00 © 2007 IEEE

TABLE I

PROPERTIES OF DISTRIBUTABLE ML TECHNIQUES.

ML technique	Memory reqs	Comp. reqs	Tol. to topology changes	Opt. of results	Init. costs	Add. costs
Reinforcement learning	medium	medium	high	high	medium/high	low
Swarm intelligence	medium	medium	high	high	high	medium
Heuristics	medium	low	medium	medium	high	low
Mobile agents	low	low	high	N/A	low	medium

At each step, it selects some possible action and receives a reward from the environment for this specific action. Note that the *best* possible action at some state is never known a-priori. Consequently, the agent has to try many different actions and sequences of actions and will learn from its experiences.

Usually, reinforcement learning tasks are described as a Markov Decision Process (MDP), consisting of an agent, set of possible states S, set of possible actions $A(s_t)$ for all possible states s_t and a reward function $R(s_t, a_t)$, specifying the environment reward to the agent's selected action. Additionally, the *policy* π_t defines how the learning agent behaves at some time-step t. The optimal policy is usually defined as $\pi*$. The *value function* $V(s_t, a_t)$ defines the expected total reward when taking action a_t in state s_t, if from the next state s_{t+1} the optimal policy $\pi*$ is followed. This is the function we want to learn in order to achieve the optimal policy.

RL is well suited for distributed problems, like routing. It has medium requirements for memory and computation at the individual nodes (see Table I), arising from the need of keeping many different possible actions and their values. It needs some time to converge, but is easy to implement, highly flexible to topology changes and achieves optimal results.

1) Q-learning: One simple and though powerful RL algorithm is Q-learning [5]. It does not need any model of the environment and can be used for online learning of the value function of some RL task, referred to as the Q-value function. Q-values are updated as follows:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \phi \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]$$

Where $Q(s_t, a_t)$ is the current value of state s_t, when action a_t is selected. The algorithm works as follows: at some state s_t, the agent selects an action a_t. It finds the maximum possible Q-value in the next state s_{t+1}, given that a_t is taken and updates the current Q-value. The discounting factor $0 < \phi < 1$ gives preference either to immediate rewards (if $\phi \ll 1$) or to rewards in the future (if $\phi \gg 0$). The learning constant $0 < \alpha < 1$ is used to tune the speed of learning to achieve better convergence and stability of the algorithm.

Q-learning has been already widely applied to the wireless ad-hoc scenario, for example for routing in WSNs [6]–[9]. It is very easy to implement and has a good balance of optimality to memory and energy requirements.

2) Dual RL: Dual RL is very similar to Q-learning. However, the reward function uses the best Q-values of the next state *and* the previous one. This increases slightly the communication overhead, but speeds up learning. The DRQ-Routing [10] protocol is based on it and optimizes point-to-point routing. However, compared to approaches with Q-learning, the implementation is more complex.

3) TPOT Reinforcement Learning: Team-partitioned, opaque-transition reinforcement learning (TPOT-RL) has been developed for simulated robotic soccer [11]. It allows a team of independent learning agents to collaboratively learn a shared task, like soccer playing. It differs from traditional RL in its value function, which is partitioned among the agents and each agent learns only the part of it directly relevant to its localized actions. Also, the environment is *opaque* to the agents, which means that they have no information about the next possible actions of their mates or their goodness.

TPOT-RL is fully distributed and achieves good results. However, its implementation is not trivial and it requires additional communication cost, since rewards are sent to the agents only after concluding the task (in the network context: after delivering the packet at the destination). It has been implemented for routing in TPOT-RL-Routing [12].

4) Collaborative RL: A formal definition of RL in a distributed environment and a learning algorithm is given in [13]. It presents a reinforcement learning algorithm, designed especially for solving the point-to-point routing problem in MANETs. Collaborative RL (CRL) is greatly based on Q-learning, but uses also a decay function (very similar to pheromone evaporation in Ant Colony Optimization (ACO), see Section II-B.1) to better meet the properties of ad-hoc networks. The approach is feasible also for WSNs, since it requires only minimal cost overhead.

B. Swarm Intelligence - SI

The term Swarm Intelligence refers to a class of machine learning techniques, biologically inspired by the behavior of social insects like ants, bees etc. The main idea is the distributed nature of the algorithms, where individual agents have only very limited memory and computational resources. However, the agents are able to communicate to each other through the shared environment (like ants' pheromone trails) and to learn cooperatively its properties.

Swarm intelligence is well suited for distributed network scenarios, where mobility and topology changes are of greatest importance, but energy is not restricted, like MANETs (see also Table I). Like Reinforcement Learning, Swarm Intelligence techniques need some reasonable amount of memory and computational resources on the network nodes and is very adaptable to topology changes. The results under perfect network properties are optimal. Good introduction into Swarm Intelligence for wireless communications is presented in [14] and more detailed information is given in [15].

1) Ant Colony Optimization (ACO) and AntNet: Ant Colony Optimization was first introduced by Marco Dorigo in [16], [17]. The algorithm finds near-optimal solutions to graph optimization problems. Ants walk on the edges of the graph, leaving pheromones on their way, which is used to optimize the paths of future ants.

AntNet [18] is the ACO application in communication networks and is used to find near-optimal routes in a communication graph without global information. The agents are divided into *forward* and *backward* ants. Forward ants are initialized at the data source and sent to all known destinations at regular intervals, travel through the network graph *randomly* choosing the next hop and leave pheromones on their way. The pheromones are saved in special pheromone tables at each network node, which indicate the pheromone level for different next hops – the more ants have chosen the same path the higher the level. During their travel, forward ants gather routing information, using the arrival time at each node on their way. At destination arrival, the forward ants are transfered into backward ants and use the cashed route they have traveled to traverse it again and to update the pheromone tables according to the gathered routing information. A decay function is implemented as evaporation of the pheromone levels, indicating which routes are the most freshly used ones. Details can be found in [18], [19].

AntNet has one important disadvantage when applied to WSNs and MANETs - it causes *additonal* communication overhead by the regular sending of the ants. This is still bearable for MANETs, as AntHocNet [19] shows, but very energy-wasting for WSNs.

2) Ant-Based Control (ABC): Ant-Based Control [20] is similar to AntNet in many aspects, but also has some important differences. There are only one class of ants, started at regular intervals at the data sources, traversing the network probabilistically and updating the routing tables as they travel to the destinations. Once reaching their destination, the ants are eliminated. The update of the routing tables is thus based not on the trip times to the destination, but rather on the present *lifetime* of the ant, calculated as the delay from its launching node to the present one.

Because of its relatively smaller communication overhead (only forward ants), ABC is better suited for energy-restricted scenarios like WSN. However, it is still costly and the advantages of using it should be carefully evaluated.

C. Mobile agents.

Mobile agents are often mistaken for a Machine learning or Swarm Intelligence approach. However, they refer to the usage of simple, small entities (packets), which traverse the system (in our case the network) and deliver fresh information to the system's nodes without any communication with the environment or each other. In the case of routing (SmartAgents [21], Ant-AODV [22]), for example, the agents update paths or next hops information on the nodes. They represent a good optimization to traditional routing approaches in mobile scenarios, but increase the communication overhead.

D. Real Time Heuristic Search

Traditional heuristic search methods operate in two steps: planning and plan execution. For example, working with a search tree, they will first calculate the value function (the goodness) of all nodes and then take the best possible path through the tree. This approach cannot be applied in real time scenarios, where agents traverse the search space and have to take their decisions based on locally available data only. Real time heuristic search methods, called also agent-centered search [23] operate successfully in such environments. The agent evaluates only its current state neighborhood – the states it can reach in the next step – and executes its next action accordingly. Such an algorithm is for example LRTA* (Learning Real Time A*), where the initial values of the states are calculated using a simple heuristic (estimation of the path costs to the goal). If the used heuristic is admissible (guaranteed to never overestimate the real costs to the goal), the algorithm finds the optimal solution. More information can be found in [23], [24]. On the first glance, real time heuristic search might seem very similar to Q-learning. However, the used heuristic requires global knowledge about the environment and no exploration of non-optimal routes is ever conducted.

Real time heuristic search methods are very well suited for wireless ad-hoc scenarios and have been already applied to routing in ad-hoc networks [25], [26] with good results. However, usually a non-admissible, but easy to compute heuristic is preferred, which leads to non-optimal results. As Table I indicates, real time heuristic search methods require less computational resources and are not very flexible to topology changes.

E. Further approaches

The above presented paradigms and algorithms are well suited to be used in a distributed environment for solving different online problems, like routing in WSNs or MANETs. Some other approaches have also been used for problems where global information is available or needed, like optimal sensor nodes placement or hardware fault recognition. They include *genetic algorithms* [27], *neural networks* [28], *decision trees* and *rule learners* [29]. More details about these non-distributable techniques, their properties and advantages will be presented in a future work.

III. ML BASED AD-HOC NETWORKING PROTOCOLS AND APPLICATIONS

In this section, ML based protocols for routing in wireless ad-hoc networks (both MANETs and WSNs) will be presented. This is not an exhaustive survey, but it tries to include all important and trailblazing works in the area. Other applications like clustering, fault recognition etc. will be extensively surveyed in a future work. In order to simplify comparison, names are introduced for the protocols where necessary.

A. Routing in MANETs

AntHocNet and extensions: One of the mostly cited routing protocols, using ant colony optimization is AntHocNet [19] (see also Section refACO). It is based on AntNet [18], developed earlier by some of the same authors, but is designed especially for the needs of a wireless ad-hoc network and is the best explored and evaluated one in the literature for using swarm intelligence in wireless networks. An extension of AntNet is ARS [30], which supports also quality of service guarantees in communication networks, like bandwidth and hop-count by defining constraints over the links of the network.

MANSI: MANSI [31] is a multicast routing protocol for MANETs. It is similar to traditional multicast protocols, where a core node initiates the building of the multicast tree through a forward Join Request Packet and a backward Join Reply Packet. However, nodes different from the core send ants into the network at regular intervals to explore the network for better routes to the core, leaving routing information (pheromones) on their ways. This information is later used by following ants for opportunistically selecting their next hops. The approach is similar to AntHocNet [19], however, optimization is applied to multicast instead of unicast routing. The approach is well described and achieves good results. However, it is not applicable to WSNs, since it requires a great communication overhead for constructing the tree and optimizing the routes via ants.

UniformAnts: [32] presents a simple ant-optimization based technique for finding and maintaining routes in a wireless network. It uses only forward ants, updating the probability-based routing tables on the nodes as the ant travels towards the sink. As any other ant-based approach, it sends ants at regular intervals through the network which causes a fairly high communication overhead. However, sending only forward ants restricts somehow the additional communication costs and makes the approach better suited for WSNs.

Q-MAP: Multicast routing has been addressed also by the authors of [9]. They use a Q-learning approach to find and build the optimal multicast tree in a MANET. The protocol consists of the two traditional multicast building phases: a join query forward propagation for finding the best route and a join reply backward to form the optimal route. Q-values are associated with different upstream nodes and the best Q-values are disseminated directly from the sinks to the source during the second phase, thus making exploration of routes unnecessary and speeding up the process. However, this exploration-free learning not only stays in conflict with the learning paradigm, but is also insensitive to topology changes and reduces the protocol to a static approach.

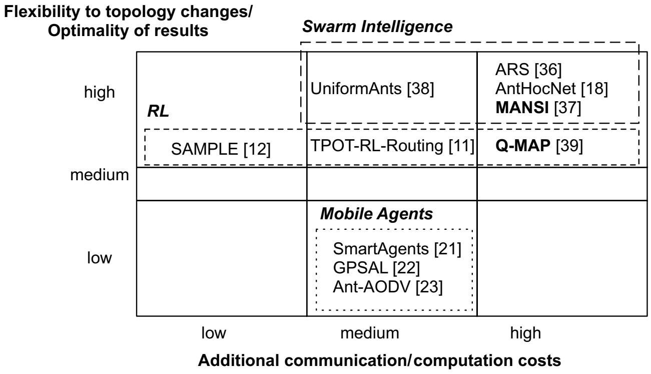

Fig. 1. Properties of existing ML based routing protocols in MANETs. Multicast protocols are given in *italic*, protocols based on the same ML technique are surrounded by a dashed box.

TPOT-RL-Routing: TPOT-RL (see Section II-A.3) is applied to packet routing in a network in [12]. The goal of the paper is a proof-of-concept of the wide applicability of the algorithm rather than developing a high-performance routing protocol and thus does not provide any comparison to other approaches. Besides this, although application to network routing is possible, it is not the best representation of the problem for two reasons: first, it presents the network as *one* system with a large number of possible states, and second, it assumes that additionally for every packet sent to the sink, a backward packet is sent also back to the *sender* to compute how many hops it travelled and thus to deliver some reward to the learning agent. Both assumptions are valid, but do not optimally use the properties of a network (wireless or not).

SAMPLE: Collaborative RL (see Section II-A.4) is defined and applied to optimal routing in a MANET in [13]. The approach learns the optimal routing policy through feedback among agents. A routing protocol is implemented on top of CRL, called SAMPLE, and tested in different network topologies with various mobility. The approach is fully distributed and the implementation is feasible also for WSNs, since all routing information is sent together with the data packets.

Summary of works: Existing ML-based routing protocols in MANETs usually use one of the following three techniques: reinforcement learning, swarm intelligence or mobile agents. The different properties of the resulting protocols can be clearly identified from the summary, given in Figure 1. Swarm intelligence, for example, causes higher communication overhead by the sending back and forth many learning agents (ants), but achieves optimal results also in a highly mobile environment. Thus, this technique could be considered the first choice when acting in a MANET, where high mobility is present but energy is not restricted.

Reinforcement learning, on the other hand, does not require higher communication costs in the usual case where routing information is sent together with the data packets. However, this means also that routing information is disseminated at most at the same speed as data is forwarded in the network. In case of low data workload, routing information will get either old or will be disseminated only *after* data is sent. This is not well suited for scenarios with high required quality of service, like multimedia applications. Another possibility, not found in any of the existing protocols, will be to separate the data from the control packets and to achieve better flexibility through constantly exploring the network. However, this resembles already too much a swarm intelligence approach.

Properties of mobile agents based approaches are given for comparison.

B. Routing in WSNs

Q-Routing: One of the fundamental and earliest works in packet routing using machine learning techniques is Q-Routing [7]. The authors describe a very simple, Q-learning based algorithm, which learns the best paths considering the least latency to the destinations. Simulations proved the algorithm to be highly efficient under high network loads and to perform also well under changing network topology. Although the approach was developed for wired, packet-switched networks, it can be easily applied to wireless networks and is especially well suited for WSNs, since it uses only locally available information at the nodes and is fully distributed.

DRQ-Routing: DRQ-Routing [10] is based on Q-Routing and uses the same WSN application scenario: routing packets from a source to a sink, while minimizing delivery time of the packets. However, the authors use dual reinforcement learning (see Section II-A.2). Thus, learning converges faster and the protocol shows better performance. The approach is again fully distributed and uses only local information and feedback appended to packets from neighboring nodes. However, its communication cost is slightly increased by the backward rewards, compared to Q-Routing.

Q-RC: Q-RC (Q-Routing with Compression) [6] presents a routing protocol, where the goal is to aggregate the source packets as early as possible in the path, compress them and send to a single sink. The best compression path is learned by Q-learning. The approach is fully distributed and can be applied easily to similar routing problems. However, the protocol is somewhat premature, since it gives no communication details, nor an implementation for exchanging the rewards.

FROMS: We use also a simple Q-learning approach in our multicast protocol, called FROMS (Feedback Routing for Optimizing Multiple Sinks) [8]. Its goal is to route data efficiently from one source to many mobile sinks in a WSN. The approach uses Q-learning to incrementally learn the real costs of different possible shared routes (trying also suboptimal routes). The algorithm is fully distributed and can be easily applied to a wide range of problems in WSNs.

SARA: The work in [26] proposes a family of statistically assisted routing protocols for routing from one source to a single destination. Besides some energy-based greedy forwarding techniques, the authors propose an additional, learning-based algorithm. It requires an exact estimation of the real path costs and allows cost information to propagate through the network. For that, LRTA* is used to learn the cost estimations through online heuristic search. The algorithm is fully distributed and well theoretically founded. However, a real world protocol is missing as well as implementation details of the cost propagation or overhead estimations.

CB-LRTA:* A variation of LRTA* is introduced by [25], where a simple routing protocol is developed, called CB-LRTA* (Constraint-Based LRTA*). It uses a set of constraints, defined on connections and nodes to find the best path to some destination using online heuristic search. Once the packet arrives at the destination, routing information is conveyed the whole way back to the source, updating the learned next hop routing data at the intermediate nodes. The approach is interesting and novel, since it allows a wide variety of different tasks to be defined in the network for routing. However, its applicability to the WSN domain is questionable since it requires a back traversing of the paths in order to learn the best ones. Additionally, the presented protocol is not mature and only a basic idea of its performance and overhead is given.

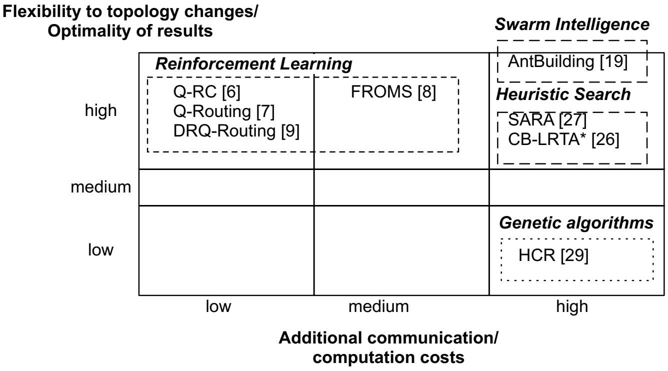

Fig. 2. Properties of existing ML based routing protocols in WSNs. Multicast algorithms are given in *italic* , protocols based on the same ML technique are surrounded by a dashed box.

HCR: Hierarchical Cluster Based-Routing (HCR) [33] is an extension of the well known LEACH [34] clustering algorithm, which uses additionally a genetic algorithm to form the clusters. It is assumed that the base station has complete knowledge of the network - topology and battery status of all nodes. It then uses a genetic algorithm to compute the best clusters - members and cluster heads - for this networks and sends a broadcast with the full cluster information to all nodes in the network (wide-range one-hop broadcast). The algorithm has several severe drawbacks - first, it drains the batteries of cluster heads too fast and inefficiently and second, and more important, it assumes global knowledge about the network topology and cannot handle topology changes or asymmetric links. Using a genetic algorithm is also unsuitable, since it is time- and computation-intensive.

Summary of works: A summary of the properties of all presented routing protocols in WSNs in given in Figure 2. The used machine learning approaches can be easily concluded from the properties of the protocols: for example, RL based methods have low computational and communication requirements and achieve good results both in terms of finding an optimal solution and keeping it easily in the presence of topology changes (flexibility). The only exception of this rule is FROMS [8], because of the slightly different problem of routing to *multiple* sinks. This makes the problem harder and the number of possible actions (next hops) at each node increases. Consequently, the computation costs increase too and the learning phase of the protocol is longer.

Swarm intelligence and heuristic based methods have one major disadvantage: they require a back propagation of the routing costs back from the destination to the source, thus doubling the communication cost per data packet. Still, they achieve very good results in terms of optimality and flexibility.

On the other hand, centralized approaches like genetic algorithms [27] are inapplicable to routing in WSNs: they require global position and topology information, thus causing a great communication overhead. Additionally, they cannot adapt to topology changes (the solution has to be recomputed from the beginning) and cannot deal with communication problems like link asymmetry and packet losses. Given the global topology, they are able to compute an optimal or near-optimal solution to the problem (like optimal clusters). However, their inflexibility outmatches the optimal results they achieve (see Figure 2).

IV. Summary

In this work, a survey of the most important ML-based solutions to the routing problem in wireless ad hoc networks (WSNs and MANETs) is presented. Additionally, a guide on machine learning techniques and their applicability to the described domain is given. Future work will extend the list of presented ML techniques and networking applications.

Machine learning has been applied to routing problems in wireless networks as early as 1994 [7] and since then finds more and more applicability. Special distributed algorithms have been developed [13] or adjusted [11] to the wireless ad hoc scenario and a large spectrum of traditional ML approaches has been applied to different problems, from reinforcement learning and swarm intelligence to neural networks.

The quality of the works and their mostly good results compared to non-learning protocols clearly show the wide applicability of ML approaches and their advantages in a mobile, unreliable environment. Additionally, several ML techniques have emerged to be the most appropriate for use in routing for wireless ad hoc networks: reinforcement learning for energy-restricted mostly static WSNs and swarm intelligence for highly mobile, non-energy restricted MANETs. Both are fully distributed, highly tolerant to environmental changes and easy to implement. On the other hand, only a few of the protocols have reached a more mature phase and have been tested in a real deployment or have been extensively evaluated against various other solutions, both traditional and ML-based. Also, in some cases inappropriate ML algorithms have been used, leading to higher communication costs or lower quality results.

In summary, future research in the area has to concentrate on further development and adjustment of the algorithms to the particular scenario and leading the solutions to a mature protocol state with real world implementation. Additionally, ML approaches should be applied to a broader set of problems and protocol stack layers, with the goal of more flexibility and independence of the systems.

References

[1] K. Roemer and F. Mattern, "The design space of wireless sensor networks," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 54–61, 2004.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[3] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, March 1998.

[4] L. P. Kaelbling, M. L. Littman, and A. P. Moore, "Reinforcement learning: A survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[5] C. Watkins, "Learning from delayed rewards," Ph.D. dissertation, Cambridge University, Cambridge, England, 1989.

[6] P. Beyens, M. Peeters, K. Steenhaut, and A. Nowe, "Routing with Compression in WSNs: A Q-Learning approach," in *Proc. of the 5th Eur. Wksp on Adaptive Agents and Multi-Agent Systems (AAMAS)*, 2005.

[7] J. A. Boyan and M. L. Littman, "Packet routing in dynamically changing networks: A reinforcement learning approach," *Advances in Neural Information Processing Systems*, vol. 6, 1994.

[8] A. Egorova-Förster and A. L. Murphy, "A Feedback Enhanced Learning Approach for Routing in WSN," in *Proc. of the 4th Wksp on Mobile Ad-Hoc Networks (WMAN)*. Bern, Switzerland: Springer-Verlag, 2007.

[9] R. Sun, S. Tatsumi, and G. Zhao, "Q-map: A novel multicast routing method in wireless ad hoc networks with multiagent reinforcement learning," in *Proc. of the IEEE Conf. on Comp., Comm., Control and Power Engineering (TENCON)*, vol. 1, 2002, pp. 667–670 vol.1.

[10] S. Kumar and R. Miikkulainen, "Dual reinforcement q-routing: An on-line adaptive routing algorithm," in *Proc. of the Artificial Neural Networks in Engineering Conf.*, 1997.

[11] P. Stone and M. Veloso, "Team-partitioned, opaque-transition reinforcement learning," in *Proc. of the 3rd Annual Conf. on Autonomous Agents (AGENTS)*. New York, NY, USA: ACM Press, 1999, pp. 206–212.

[12] P. Stone, "Tpot-RL applied to network routing," in *Proc. of the 17th Int. Conf. on ML*. San Francisco, CA: Morgan Kaufmann, 2000.

[13] J. Dowling, E. Curran, R. Cunningham, and V. Cahill, "Using feedback in collaborative reinforcement learning to adaptively optimize manet routing," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 35, no. 3, pp. 360–372, 2005.

[14] I. Kassabalidis, E. M. A. Sharkawi, R. J. Marks, P. Arabshahi, and A. A. Gray, "Swarm intelligence for routing in communication networks," in *Proc. of the IEEE Global Tel. Conf. (GLOBECOM)*. IEEE Press, 2001.

[15] J. Kennedy and R. Eberhart, *Swarm Intelligence*. Morgan Kaufmann, 2001.

[16] M. Dorigo, "Optimization, learning and natural algorithms," Ph.D. dissertation, Politecnico di Milano, Italy, 1992.

[17] M. Dorigo and T. Stuetzle, *Ant Colony Optimization*. MIT Press, 2004.

[18] G. Di Caro and M. Dorigo, "Antnet: Distributed stigmergetic control for communications networks," *Journal of AI Research*, vol. 9, pp. 317–365, 1998.

[19] G. Di Caro, F. Ducatelle, and L. Gambardella, "AntHocNet: an adaptive nature-inspired algorithm for routing in mobile ad hoc networks," *Eur. Trans. on Telecommunications*, vol. 16, pp. 443–455, 2005.

[20] R. Schoonderwoerd, O. Holland, J. Bruten, and L. Rothkrantz, "Ant-based load balancing in telecommunications networks," *Adaptive Behavior*, no. 2, pp. 169–207, 1996.

[21] E. Bonabeau, F. Henaux, S. Guérin, D. Snyers, P. Kuntz, and G. Theraulaz, "Routing in Telecommunications Networks with "Smart" Ant-Like Agents," in *Proc. of the 2nd Int. Wksp on Intelligent Agents for Telecommunications Applications (IATA)*, Paris, France, 1998.

[22] C. Tham, S. Marwaha, and D. Srinivasan, "Mobile agents based routing protocol for mobile ad hoc networks," in *Proc. of the IEEE Global Telecommunications Conf. (GLOBECOM)*. IEEE Press, 2002.

[23] S. Koenig, "Agent-centered search," *AI Magazine*, vol. 22, no. 4, pp. 109–131, 2001.

[24] R. E. Korf, "Real-time heuristic search," *Artificial Intelligence*, vol. 42, no. 2-3, pp. 189–211, 1990.

[25] Y. Shang, M. P. J. Fromherz, Y. Zhang, and L. S. Crawford, "Constraint-based routing for ad-hoc networks," in *Proc. of the Int. Conf. on Information Technology: Research and Education (ITRE)*, 2003.

[26] M. Rossi, M. Zorzi, and R. R. Rao, "Statistically assisted routing algorithms (SARA) for hop count based forwarding in wireless sensor networks," *Wireless Networks*, June 2006.

[27] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Prentice Hall International, 2003.

[28] R. Rojas, *Neural Networks - A Systematic Introduction*. Springer-Verlag, 1996.

[29] T. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[30] K. Oida and M. Sekido, "An agent-based routing system for qos guarantees," in *Proc. of the IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC)*, vol. 3. IEEE Computer Society, 1999, pp. 833–838.

[31] C.-C. Shen and C. Jaikaeo, "Ad hoc multicast routing algorithm with swarm intelligence," *Mobile Networks and Applications*, vol. 10, no. 1-2, pp. 47–59, 2005.

[32] D. Subramanian, P. Druschel, and J. Chen, "Ants and reinforcement learning: A case study in routing in dynamic networks," in *Proc. of the 15th Joint Conf. on AI (IJCAI)*. The MIT Press, 1997.

[33] A. W. Matin and S. Hussain, "Intelligent hierarchical cluster-based routing," in *Proc. of the Int. Wksp on Mobility and Scalability in WSNs (MSWSN)*, San Francisco, CA, 2006.

[34] W. Rabiner-Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," in *Proc. of the 33rd Hawaii Int. Conf. on System Sciences*, Washington DC, USA, 2000.