Ann. Telecommun. (2017) 72:3–18
DOI 10.1007/s12243-016-0533-8

Information-centric sensor networks for cognitive IoT: an overview

Fadi M. Al-Turjman [1]

Received: 31 October 2015 / Accepted: 20 June 2016 / Published online: 4 July 2016

Abstract Information-centric sensor networks (ICSNs) are a paradigm of wireless sensor networks that focus on delivering information from the network based on user requirements, rather than serving as a point-to-point data communication network. Introducing learning in such networks can help to dynamically identify good data delivery paths by correlating past actions and results, make intelligent adaptations to improve the network lifetime, and also improve the quality of information delivered by the network to the user. However, there are several factors and limitations that must be considered while choosing a learning strategy. In this paper, we identify some of these factors and explore various learning techniques that have been applied to sensor networks and other applications with similar requirements in the past. We provide our recommendation on the learning strategy based on how well it complements the needs of ICSNs, while keeping in mind the cost, computation, and operational overhead limitations.

Keywords Information-centric sensor networks · Cognitive node · QoI · Reinforcement learning

1 Introduction

Wireless sensor networks (WSNs) have evolved from simple sensing and tracking applications to being an integral and essential part of the Internet of things paradigm. This means that sensor networks have to deal with large amounts of data, support requests from multiple users, and support information extraction from the network rather than serving as point-to-point communication networks that transmit data from a source to sink. To enable WSNs to easily integrate with, and adapt to the IoT environment [1], we propose the use of learning, as an element of cognition in the network. Cognition refers to the ability to be aware of the environment, be able to learn from the past actions, and use it to make future decisions that benefit the network [2]. Learning is one of the elements of cognition that can achieve different goals in different systems. In robotic chess, learning can be used to *plan* moves based on opponents' actions; in aircraft autopilot systems, learning can be used to *control* the plane's navigation; and in cognitive networks, learning can be used to improve *decision making* that improves network management and its overall performance. Whatever be the system's goal, the performance of the learning technique depends on three main tasks: (1) observations made from current activities in the environment, (2) feedback from past actions, and (3) how this information is used to achieve the system's goals. An information-centric sensor network (ICSN) has specialized nodes called *cognitive nodes* that are capable of performing all these tasks by implementing major cognition elements. These major elements are the *learning*, *reasoning*, and *knowledge representation*. Figure 1 represents these three major elements for cognitive nodes in ICSN, and associate them with their respective functions. These elements of cognition, when incorporated in the network nodes of a WSN, help in providing better understanding and catering for the end user requirements. Our expectation from the cognitive elements would be to cater to the following objectives. On the short term, to observe current network behavior and respond adaptively to changing network dynamics. And on the long term, to learn from the previous behavior and plan better for the future so as

✉ Fadi M. Al-Turjman
fadi@metu.edu.tr

[1] Department of Computer Engineering, Middle East Technical University, Northern Cyprus Campus, 99738 Kalkanlı, Güzelyurt, Mersin 10, Turkey

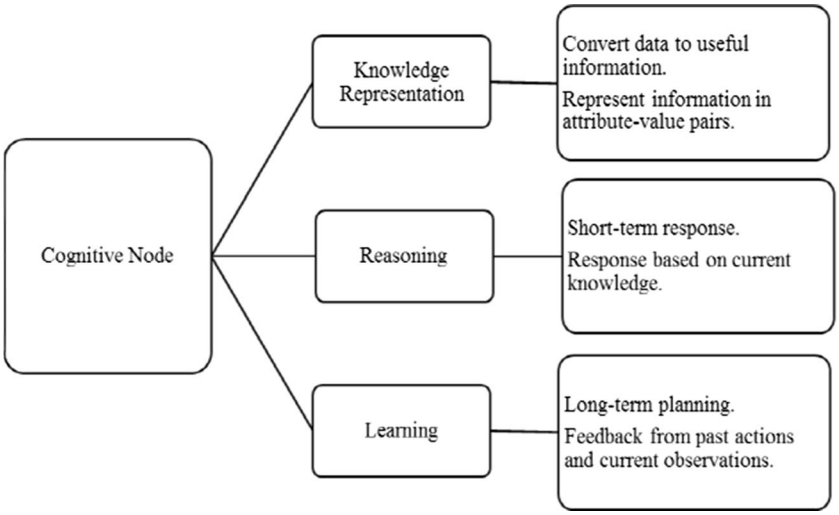

Fig. 1 Cognitive node and its elements

to make predictions and decisions that positively impact the network survivability and application QoI during its lifetime. Using these elements, a conceptual architecture of the cognitive node can be illustrated as detailed in Fig. 2.

This cognitive sensor network framework is mainly targeted towards application such as smart cities and smart outdoor monitoring in IoT era. In these applications, the goal for the learning algorithm is to dynamically adapt routing decisions to improve the quality of information (QoI) delivered to the user [3] and improves the network lifetime. This can be achieved by using a learning algorithm that can correlate the impact of past decisions on the current network behavior, based on end user's requirements. But these are challenging applications for the learning algorithm due to the large scale of the network, changing network topology due to node deaths and varying channel conditions, and heterogeneous traffic flows generated as a result of changing user requests in the IoT application domain. For the learning algorithm to be successful in such a dynamic environment, it must occur incrementally and span over multiple episodes [4]. Only then will it be able to adapt to the irreversible changes in the environment and contribute towards delivering QoI-aware data to the end user over the network's lifetime. In this paper, we explore the different classes of learning techniques and identify what works best for large-scale information-centric sensor networks in IoT applications.

The remainder of the paper has been organized as follows: In section II, we look at some WSN design issues in IoT applications, and summarize the design changes required for integration of WSNs with IoT. In section III, we explore artificial intelligence and learning used by WSNs, to better analyze their suitability for ICSN applications. Subsequently, we delve into the details of ICSNs as a hybrid solution platform for integration of WSNs in IoT using a learning paradigm in section IV. We also identify a suitable learning strategy in this section, before

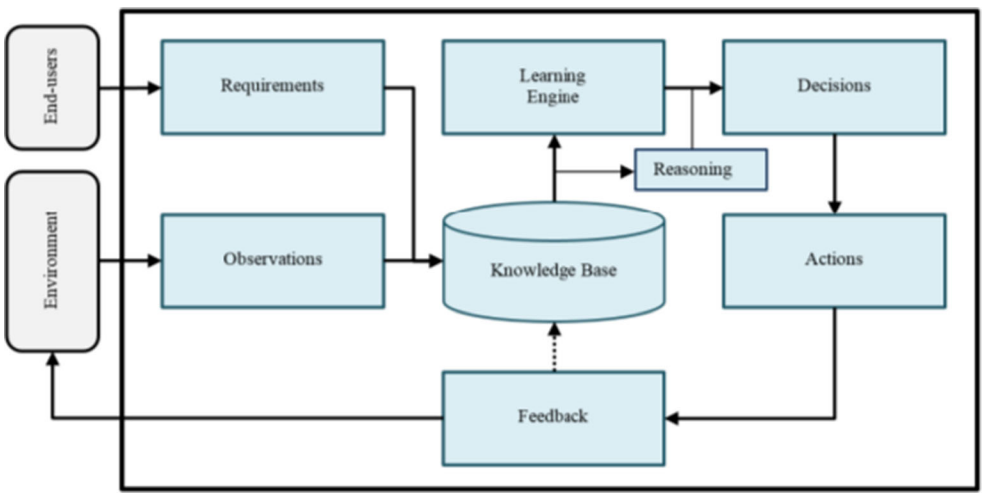

Fig. 2 Cognitive node conceptual architecture

discussing some open issues and concluding the paper in section V.

2 WSN design issues in IoT applications

In this section, we take a brief look at the various design issues that need to be addressed to seamlessly integrate WSNs in IoT applications. We categorize the design issues into two parts: (i) expectations of the users from the network, which includes feature requirements of the sensor network's interface with the access network (future internet) and (ii) adaptations and changes required within the sensor network to cater to user requests while managing network resources. Each of these design issues will be explored in the sub-sequent sections, and we will see why these issues need to be handled differently from existing WSN applications.

2.1 User expectations from the network

Traditional WSNs were designed for specific applications such as target tracking, temperature monitoring in a building, and movement of goods in a supply chain, to name a few. Users accessed the network only when they needed a particular type of sensed value, such as temperature, pressure, or humidity for instance. However, in the IoT era, sensor nodes have become heterogeneous and are capable of supporting multiple types of sensors. This way the sensor network can be expected to support multiple applications and provide users with a variety of data as supported by the type of sensor nodes used. Thus, WSN applications should evolve from supporting application-specific deployments to providing an application platform that users can access to gather a variety of data [5].

2.1.1 Multi-user application platform support

The basic idea behind developing an application platform is to provide a flexible, generic infrastructure that can lead to easy adoption of WSNs into a variety of IoT applications. For example, a sensor network deployed in a city should be capable of providing data for the following applications: (1) air pollution monitoring; (2) daily weather monitoring (temperature, humidity, UV index); and (3) park and garden irrigation management. Such an application platform and its associated services would also support the conceptual ubiquitous sensor network (USN) used in large-scale sensor network deployments [6, 7]. This would invite more number of users to simultaneously access the network, which makes the WSN design even more complex. The underlying sensor network will have to support heterogeneous traffic flows generated as a result of simultaneous access from multiple users, which is a very challenging task for the resource-constrained sensor network.

2.1.2 User requirement—aware request classification

In a large-scale deployment of sensor networks that allows multiple users to access it, different users may have different requirements on the desired quality of experience (QoE) [8], and the network may have its own limitations on the quality of service (QoS) it is able to support [9–11] . While the user requirements may evolve over time, the sensor network gradually decays in terms of its energy capacity, and it also involves dynamic changes in the link conditions and node availability. In addition, the user's expectations from the attributes associated with delivered data also vary based on the application and type of request. Hence, there is a need to monitor data attributes from the user requirements perspective directly, a skip level from the application interface, as shown in Fig. 3. Latency, reliability, accuracy, relevance, and robustness are some of the attributes that can collectively provide an estimate of the quality perceived by the user based on the information received at the user-end from the network. This is referred to as the quality of information (QoI) metric, and it may provide information about the success of the network in satisfying the evolving user requirements, while simultaneously saving valuable network resources such as bandwidth and energy [12]. Thus, it becomes necessary for the underlying sensor network to be aware of user requirements and be able to classify user requests to deliver data in compliance with the desired QoI. While there has been research in the area of QoI assurance based on changes in the phenomena observed in the network [13], there has not been much work on the user interface side for request classification based on changing user requirements.

2.1.3 Internet access interface between user and WSN

ZigBee-based address-centric sensor networks facilitate the integration of WSNs with the Internet, as both the networks are address-centric and point-to-point. However, recent advances in the future internet research suggest that an information-centric approach to data delivery is favored over the address-centric approach. This is because researchers believe that it will become increasingly difficult to handle the growing IP address space that serves billions of users in the near future. Information-centric network (ICN) is one of the promising architectures for the future internet paradigm [14]. It proposes an information-centric approach to data access

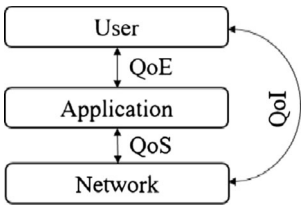

Fig. 3 QoI-aware data delivery in WSNs for IoT applications

where users look for named data objects instead of looking up IP addresses to find data they are interested in. The architecture is set up to support data storage at strategic locations in the network, typically the edge of the network, so that requests do not have to be propagated deep into the network to access the required information. If the user had to request for data through such an information-centric interface, then the network interface would also have to be modified to ease the flow of information. Hence, for IoT applications to be able to adapt to changes of the future internet paradigm, changes would be required in the way requests are made at the user access interface and also at the sensor network interface.

2.2 Adaptations at the network level

In this section, we move from the issues at the user interface, to the design issues at the network level to identify the adaptations required to enable WSNs for IoT applications.

2.2.1 Energy considerations and resource management

Large-scale platform-based sensor network deployments that are accessed by multiple users tend to benefit more from a data-centric approach than an address-centric one. For instance, a user may request for information such as: temperature readings from all regions in the network where value is greater than 25 °C. In such a scenario, there is no specific address that the user is requesting data from. Instead, the query is information-centric and requires information from the entire network. This could be very energy intensive if appropriate query dissemination and data delivery techniques are not identified. In their work on energy conservation schemes in WSNs, Anastasi et al. have extensively explored data-driven techniques, broadly classified into data reduction and energy efficient data acquisition approaches [15]. Several researchers have also established the energy conserving nature of data-centric sensor networks [16–18]. However, previous research has only considered networks of a few hundreds of sensor nodes. In IoT-based applications, there may be thousands of sensor nodes to gather information from, which adds to the complexity of the energy conservation problem. Although a data-centric approach can provide valuable energy savings in the network, further research is required to device techniques to manage information flow in such a large-scale, energy-constrained network.

2.2.2 Query dissemination and data delivery

WSNs can be broadly classified into address-centric and data-centric networks. This classification is based on how query dissemination, data gathering, and routing happen in the network. Address-centric sensor networks are built on top of the more recent ZigBee protocol [19] that provides a service-oriented framework for implementing WSN applications. Data is routed using the tree-based hierarchical topology consisting of router and coordinator nodes, while sensor nodes are the sources of information. Routers off-load the sensor nodes by carrying forward their data to the sink, thus bringing considerable energy savings to the battery-operated sensor nodes. However, WSNs are essentially information extraction networks. They were originally developed as data-centric sensor networks (DCSNs) that did not make use of node addresses. Instead, their focus was on the attributes of the requested information, which was gathered from wherever it was available in the network, and delivered to the sink. Handling query dissemination and delivering the gathered data is a very challenging task in large-scale sensor networks. This is because of (1) the ad hoc nature of the wireless channel, (2) dynamic topology changes in the network due to node deaths and their changing associations, and (3) due to the nature of node distribution in the network. Hence, choosing the right approach for handling data flows in a very critical design decision, this must be made keeping in mind the interface access network (the future internet paradigm).

In addition, IoT applications have multiple users requesting for different types of data with different service requirements. For example, while request for periodic data may relax the service requirements on latency, on-demand data needs to be provided quickly, and should be relevant. On the other hand, emergency reporting must be done accurately, reliably, and quickly, with minimal delay. Thus, the way the network is setup itself will have to be modified, to minimize energy consumption during each of the phases of query dissemination, data gathering, and data delivery. Sensor node scheduling also becomes an important issue to be addressed, so that data is available when user requests for it. Moreover, planned scheduling of sensor node wake-up and sleep cycles will add to the energy savings and prolong the network lifetime. In addition, planning the deployment of router nodes to increase the multi-hop communication range is another aspect that needs to be considered during network design and deployment. Since the complexity of the tasks to be handled by WSNs in the IoT paradigm is quite high and multi-dimensional, it seems appropriate that the design changes consider the addition of advanced nodes [20] that can maintain connectivity in the network and carry data over long distances in these applications.

2.3 Summary of WSN design change requirements

Thus far in this section, we have seen the various design change requirements in existing WSNs to make them adapt to IoT applications. We summarize these requirements as follows:

- Multi-user application platform support
- Classification of user requests to deliver QoI-aware data

- Modification of the communication to make it compatible with the future internet paradigm—ICN
- Incorporate specialized nodes that can observe/learn from the interactions/feedback in the network and manage sensor node scheduling to prolong the network lifetime
- Plan the deployment of router and specialized nodes to maintain network connectivity and enable multi-hop data transmission over the large-scale network
- Consider energy efficient query dissemination and data delivery, and dynamic traffic management due to changing network conditions and user requests

Figure 4 summarizes all the design change requirements in the form of a conceptual design for the future IoT paradigm that supports multiple users and integrates ICN-based internet access, and the large-scale data-centric sensor network. We call this an information-centric sensor network (ICSN) [21–24]. Comparing DCSN protocols with the information-centric networking (ICN) approaches [25, 26] for future internet, we can see that DCSNs already implement two major features of the ICNs. Firstly, the naming schemes, or named data objects for referencing requested data instead of using node addresses, and secondly, storage of collected data in nodes for ease of access. We take these two features as a strong indication of the need to shift to data-centric sensing schemes for WSNs, but with ZigBee and the information-centric approach to adapt to the advanced applications in the IoT era [27]. Although the data-centric approach will help to better manage the network at each of the interfaces: user, access network, and sensor network, the authors do think that a simple integration of different technologies will not be adequate to manage the complexity of the tasks at the sensor network level. To improve adaptations beyond the limitations of traditional cross layer design, we look to the tools of artificial intelligence to support the ICSN paradigm.

3 Artificial intelligence and learning in WSN

Various artificial intelligence (AI) techniques have been applied to WSNs to improve their performance and achieve specific goals. We look at AI techniques as a means of introducing learning in the WSN. Learning is an important element in the observe, analyze, decide, and act (OADA) cognition loop [28, 29], used to implement the idea of cognitive wireless networks [30, 31]. In this section, we broadly classify AI techniques as computational intelligence (CI) techniques, reinforcement learning (RL) techniques, cognitive sensor networks and multi-agent systems (MAS), and context aware computing as shown in Fig. 5.

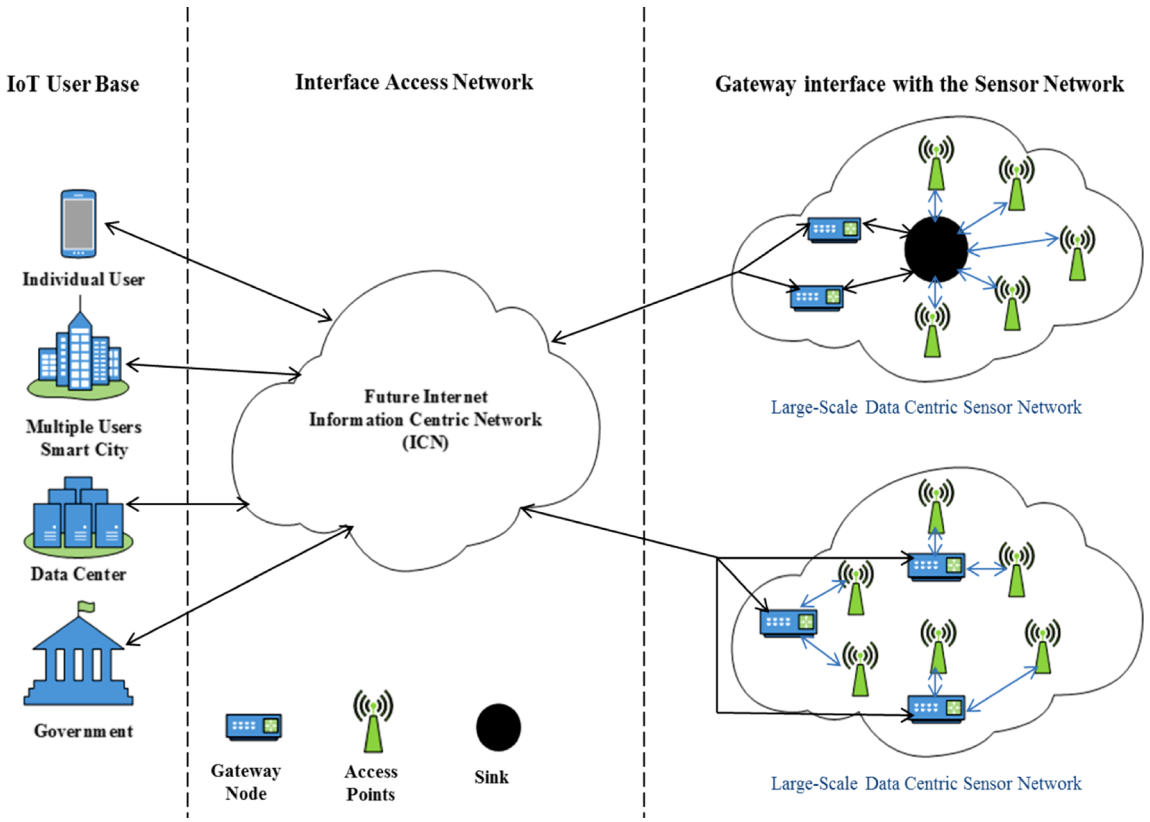

Fig. 4 Conceptual design of an ICSN for IoT applications

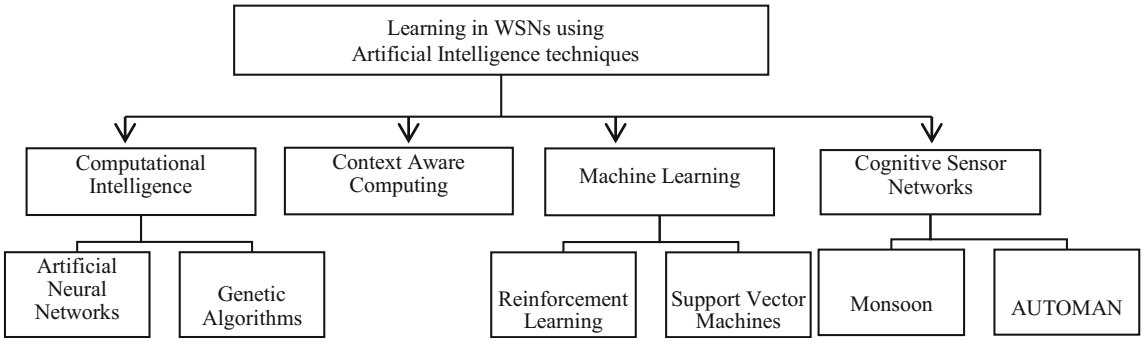

Fig. 5 Learning techniques used in WSNs

3.1 Computational intelligence

CI techniques are a set of nature-inspired computation methodologies that help in solving complex problems that are usually difficult to fully formulate using simple mathematical models. Examples of CI techniques include genetic algorithms, neural networks, fuzzy logic, simulated annealing, artificial immune systems, swarm intelligence, and evolutionary computation. In a learning environment, CI techniques are useful when the learning agent cannot accurately sense the state of its environment. In WSNs, CI techniques have been applied to problems such as node deployment planning, task scheduling, data aggregation, energy-aware routing, and QoS management. Kulkarni et al. have provided an extensive survey of CI techniques applied to WSNs [32]. They elaborate on various CI techniques and associate each with typical problem domains they can solve in WSNs. From their observations, swarm intelligence applied to solving the routing and clustering problem has drawn the most research attention in recent times. However, a major drawback of this methodology is that it can be computationally intense and may require some form of model-based offline learning to deliver to the requirements of the application scenario. Techniques such as ant colony optimization can cause an undesirable increase in communication overhead in WSNs [33] too. Apart from these drawbacks, none of the CI algorithms have been applied to solving problems of data representation, aggregation, and delivery in a distributed, decentralized setup, under dynamic communication constraints, as is the case in IoT applications. Next, we evaluate reinforcement learning strategies used in WSNs. We separate them from other CI techniques, because they are the most widely applied learning techniques in WSNs and do not categorically fall under nature-inspired learning.

3.2 Machine learning

Machine learning can be classified into supervised, unsupervised, and reinforcement learning. Supervised learning would be more compute intensive and requires a training sequence. Additionally, accuracy of the learning algorithm would then be defined by this training sequence. In the unsupervised learning approach, the learning is from the environment being observed and no training sequence is required. Reinforcement learning (RL) is a reward-based technique that emphasizes on learning while interacting with the environment, without relying on explicit supervision or complete model of the environment. It is a method of automating goal-directed learning and decision making. Since it is advantageous to be aware of, and learn the changes in the environment in WSN-based applications, RL would be an appropriate choice as the learning strategy. In WSNs, RL has been successfully applied in networking tasks such as adaptive routing, identifying low cost and energy-balanced data delivery paths [34, 35], and in information processing tasks involving data aggregation and inference [36]. In the sub-sequent sections, we explore the different types of RL methods as reward-based strategy [37].

3.2.1 Model-based reinforcement learning

A learning agent (LA) in model-based RL collects experiences and builds a model from that. The actions can be chosen randomly or heuristically and observe the impact on a model and the reward (where s is the current state, a is the actions, and s' is the next state). This means, by taking action a, how often would the LA end up in the state s' if it started from state s and estimating a probability by counting the number of times triple occurs over the sample space. With this information, the LA builds an estimate of the model and the reward by estimating probabilities based on the number of trials (or episodes). Once an estimate of the model and rewards are ready, the LA can plan its actions. A good plan can be found from policy iterations or value iterations. But for model-based RL, the hardest part is knowing the right policy to start out with, so we can build a good model.

3.2.2 Model-free reinforcement learning

In model-free reinforcement learning, the agent is free to learn from the environment by exploring it completely on its own. Agent learns from positive reinforcement it gets

for moving towards a goal and negative reinforcement for moving away from the goal. Q-learning is a form of model-free RL in which the learning agent converges to an optimal policy even if it were acting sub-optimally. This is called off-policy learning. This is the most extensively used form of RL, as it is easy to implement and a relatively low-cost solution. However, Q-learning has its limitations too. The agent has to explore enough and has to eventually make the learning rate small, but not decrease it too quickly, so that it has a large enough state space that covers all possible actions and policies. In the context of the ICSNs, we are not interested in finding the optimal policy. Rather, we are interested in any sub-optimal solution that does not take much time to converge and can act faster. It does not even matter how the action is selected, and a heuristic choice could work well too. This way, an even more simplified version of RL can be applied to ICSNs. We will discuss this in more detail in section IV.

3.2.3 Support vector machines

Support vector machines (SVMs) are a class of ML algorithms that were originally formulated for binary classification. They can be applied to complex, highly non-linear problems in a consistent and structured manner and have been successfully applied to intrusion detection and security problems in WSNs [38]. This technique has not been applied to any other design problem in WSNs, but has proved very effective in small-scale sensor networks of about 50 sensor nodes randomly deployed in a 100-m × 100-m area. This class of ML algorithms has great potential to be applied to security in IoT-based applications, especially if the interface becomes information centric, with gateway nodes lying within the sensor network itself. As sensor network becomes more vulnerable to attack in the ICSN setting, SVM techniques can be further explored to secure large-scale networks in IoT applications.

3.3 Cognitive framework and multi-agent systems

Cognitive networks are built around the idea of having sensor networks evolve around user requirements. It is about taking a step towards developing intelligent networks that do not limit themselves to point-to-point communication within the network. Instead, they enable the network to perceive user requirements and deliver data using distributed intelligence in the network. To implement distributed intelligence, multi-agent systems (MAS) are typically used. The agents in these MAS are called cognitive agents. They may interact to achieve information fusion and retrieval and may also be able to predict data for future use. Typical applications include medical monitoring, intelligent transport, and smart environment monitoring [39]. Such a distributed approach to introducing intelligent behavior in the network will be very beneficial in WSNs deployed for large-scale IoT applications. Some examples of successful software implementations of cognitive agents in sensor networks include AUTOMAN [40] and MONSOON [41].

3.4 Context aware computing

In large-scale sensor networks, a huge amount of data is generated. In order to derive useful information from raw data, context of the data plays an important part. Context awareness is even more important in the IoT era, as it enables the network to deliver relevant, user-requested data. While doing so, network resources are also conserved by extracting only meaningful information that is relevant to the requests, from the network. There are various aspects to context aware computing. They are the following: acquiring the context, context modeling, context reasoning, and context distribution [42]. Since we are more interested in the learning aspect as applicable to sensor networks, we delve a little deeper into the context reasoning aspect. Both CI- and RL-based techniques can be used to implement context learning, in addition to ontology-based and probabilistic logic models. The information obtained from these learning models can be exploited to infer information from stored data. Context awareness is very important and valuable in IoT applications, as it can add value to the large amount of data available from their applications.

All the learning techniques discussed in this section have been summarized in Table 1. We identify the limitations of existing techniques and provide the details of the solution platform based on ICSNs for IoT applications in the next section.

4 A hybrid solution platform: learning in ICSN

In this section, we will identify what the learning algorithm should actually learn, in order to support the information-centric nature of ICSNs. Table 2 presents three broad classes of solutions RL problems and classifies their features based on their relevance to learning in ICSNs.

4.1 What should the ICSN learn?

To identify what the ICSN should learn, let us identify what information is already known to the learning agent nodes. First, from our work in [22] and [43], we know that there is the upper bound on the maximum communication range for all the network nodes—sensor nodes, relay nodes, and the nodes that implement learning as a

Table 1 Learning using AI algorithms in WSNs

	Techniques used	Design problems addressed	Role of learning	Limitations
Computational intelligence techniques	Neural networks Genetic algorithms Fuzzy logic Simulated annealing	Node deployment Power assignment Routing Network management (coverage, lifetime)	Offline learning phase Learning to estimate congestion based on packet arrival rate and buffer size Buffer threshold management by learning packet delay and throughput values	Difficult to define appropriate quality measures for Pareto set approximations Poor adaptation to changing network topology due to offline learning phase High cost of operation (energy and delay)
Machine learning techniques	Reinforcement Learning Q-learning and its variants(FROMS) Support vector machines	Routing Node deployment Resource allocation Intrusion detection network security	Online learning of route costs Energy efficient data aggregation paths Outlier detection	Very application-specific design choices prevent building upon knowledge gained from learning in different applications Long time to converge to solution
Cognitive framework and multi-agent systems (MAS)	AUTOMAN (intelligent software agents) MONSOON	QoS management Surveillance applications Distributed intelligence for data collection under dynamic network conditions	Event-driven learning to capture changes in specific parameters (voltage) on the power grid in AUTOMAN Learning data delivery paths based on pheromone trails, adapting to node and link dynamics	Application-specific design Policy based design, no generic infrastructure Cost of inter-node communications among intelligent agents is high
Context aware computing (context learning)	Supervised Unsupervised Rule-based Fuzzy logic Ontology based Probabilistic logic	Activity or event recognition Unusual behavior detection Automated irrigation when soil is ‘dry’	For context-based reasoning and modeling decisions Using previously stored knowledge to infer contextual knowledge	Context modeling might be complex No validation or quality checking Each of the context learning techniques have unique limitations
ICSN	Knowledge representation Learning Reasoning	User requirements awareness and request classification QoI-aware data delivery Distributed intelligence	Predicting user requirements and request types Learning from feedback and observations in the network to improve QoI along data delivery paths Can be used for implementing context awareness by processing gathered data	Conceptual design that has not yet been validated in experiments Cognitive nodes might be more expensive compared to relay nodes Heuristically accelerated RL can help to reduce the learning time

Table 2 Classification of solutions to RL problems

	Features	Incremental learning support	Requires model of environment	Bootstrapping (learning a guess from a guess)	Limitations	Examples
Dynamic programming	• Computes optimal policies using a perfect model of the environment • Estimates of values of states are updated based on those of successor states	No	Yes	Yes	• Computationally very expensive • Not suitable for large problems	Policy improvement, policy iteration, asynchronous-DP
Monte Carlo methods	• Requires online or simulated sample of interaction with environment for state transitions • Averages the values of sample returns	Yes	No	No	• Maintaining sufficient exploration is an issue	On-policy methods, off-policy methods
Temporal difference	• Make long-term predictions about dynamic systems • Can be applied online with minimal computation	Yes	No	Yes	• No notable limitations; perform more efficiently with eligibility traces	Q-learning, actor-critic methods

part of cognition, the cognitive nodes. We will be referring to the learning agents as cognitive nodes (CN) from this point forward. So, the CNs need to be aware of the network topology changes only within their own communication range. The target area's coverage is taken care of at the time of deployment of nodes in the network. Thus, *learning about the topology changes in its local neighborhood* will help the CNs adapt their transmit power and choose a data delivery path that best manages the *nodes energy consumption*. Prolonging the CNs lifetime will in turn contribute towards *increasing the network's longevity*. Second, cognitive nodes store information in their knowledge base regarding the QoI performance of paths used in previous data delivery rounds. Routing tables are built and updated based on the information in the knowledge base. Unlike traditional routing tables that store static, end-to-end routing paths from source to destination nodes (usually the sink node), routing tables in ICSNs are not designed to be static. In fact, they are not even end-to-end paths, but are paths that show the network's current adaptation to the changes in topology and user requests. They store information about the most recent path used to deliver data from the CNs to the sink or other CNs or relay nodes. This means that the contents of the routing table at each cognitive node have to be updated on a regular basis to ensure they store the latest and best QoI paths. Thus, the important learning goal for the cognitive node from an *application QoI perspective is to learn the data delivery paths towards the sink that provide the best QoI values for each of the different types of user requests*. There is no one best path that is always used to route data. Instead, the routing choice depends on the current network topology, nature of the user request (periodic, intermittent-user specific, or emergency data), and volume of traffic generated in response to the request. If the routing table is viewed as cache storage, then as effective cache replacement strategy is required to replace old and redundant routing information with more recent and relevant information. These learning goals can be achieved in the following ways as represented in Fig. 6: (i) learning from feedback on current actions, (ii) learning by exploring the changes in the network topology, and (iii) learning (drawing inference) from past actions by

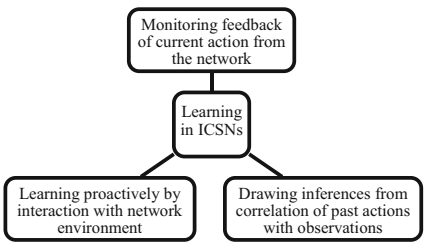

Fig. 6 Learning in sensor networks

using the information stored in a knowledge base. Once the learning converges and knowledge base updates are complete, the routing tables can be updated using information from the knowledge base. Thus, we identify two main goals of learning: Firstly, to improve network longevity and maximize the duration for which the ICSN is able to provide relevant data to the end user; and secondly, to increase the network's ability to provide QoI-aware data to the end user during its lifetime. For ICSNs, we define the network lifetime as the time after which none of the one-hop cognitive nodes to the sink are able to deliver the user's requests for information to the network, nor are they able to deliver the gathered data from the network back to the user, is called end-of-life of the ICSN.

4.2 Choosing the learning strategy

The goal of learning in ICSNs is to improve the QoI delivered to the end user and maximize the duration over which this can be maintained. In other words, learning should contribute towards maximizing the network lifetime while ensuring QoI-aware data delivery during this period. Based on these requirements, we identify the following factors that influence the choice of learning strategy in ICSNs:

- Learning should occur incrementally over time as network conditions change.
- It should support episodic learning [44] to acquire information from new events and update the knowledge base. This will let the system make decisions and act based on the observed changes.
- Learning should be distributed and occur locally when implemented in large-scale ICSNs. However, the goal of learning should be common across all the cognitive nodes.
- The learning algorithm must be light-weight and of low complexity to support episodic learning. Moreover, it must not cause too much data/control overhead in the network or negatively interfere with other nodes' functions.
- The learning algorithm should preferably be rewards based, as supervised learning would become very compute intensive for the ICSNs
- Finally, the learning algorithm must be able to converge quickly enough to be support the network's learning goals in a timely manner. It should not cost the network too much in terms of its resources (energy and time) either.

Based on these recommendations, we explore heuristically accelerated reinforcement learning techniques for use with ICSNs in the next section.

4.3 Heuristically accelerated reinforcement learning

Since both model-based and model-free RL have their limitations, we will look towards modified forms of RL solutions. We look at the possibility of using heuristics (rules of thumb) to choose a sub-optimal action instead of trying to converge at an optimal policy. The advantage of combining heuristics with RL is that RL is eventually going to converge to an optimal possible policy, but it takes time. Heuristic only attempts to make the decision choices quickly. Therefore, it will not deter the RL process, but will only enhance it to arrive at some sub-optimal solution faster, but it will not be incorrect as observed by Atkeson and Santamaria [45].

In this section, we look at the use of heuristic functions to accelerate RL algorithms. These heuristic evaluation functions called *valuation functions* are computed by stochastic sampling and dynamic programming updates [46]. The model-free method is suitable for problems that do not involve large state spaces. In contrast, the domain-independent, model-based heuristic methods can be used for solving problems with a large state space and hundreds of actions. The fast and frugal heuristics proposed by Gigerenzer [47, 48] are not only generic but are also low polynomial time and applicable to all problems that fit a given model. Another aspect to consider is that learning happens while the system is running. This makes it important to reduce the exploration space of the learning agent. In the CICSN framework, we can use the information stored in the knowledge base as a case base to choose an action that provides a close solution to the RL agent's problem.

But the choice of the case that matches the given decision problem must be chosen in a way that helps the RL algorithm to converge quickly [49, 50]. This way we can limit the exploration space by making use of the knowledge base, and use heuristics to choosen a sub-optimal action that will help the algorithm converge faster. Heuristically accelerated RL (HARL) and case-based HARL have been recently explored by Bianchi et al. in the context of robotic soccer [51, 52]. We extend this idea of accelerating RL by using heuristics and an available case base to the ICSN framework. In our application, the case base is replaced by a knowledge base that is built upon the foundation of representing knowledge in attribute-value pairs. The "Representativeness Heuristic": According to this heuristic, the more similar something is to a prototype, the more likely it is to belong to that prototype's category [53]. This heuristic is based on the fact that we tend to judge how likely something is to be true is based on how representative it is of a particular category. Thus, we conclude this section with the suggestion that heuristically accelerated RL techniques that make use of the information stored in the knowledge base of the cognitive nodes will potentially serve as low complexity solution to the learning problem in ICSNs, and might be viable in terms of the computational overhead too.

4.4 Learning in the ICSN framework

In the previous sub-section, we identified a learning strategy for ICSNs based on the network requirements and from the analysis of different learning strategies. Now we will identify how the ICSN must be set up so that the learning strategy can be implemented in this framework. We assume a deterministic deployment of relay and cognitive nodes, and number them for ease of representation.

We start with a broad classification of the type of request that an ICSN should be able to serve. We divide the requests into one of three types: type I: periodic, type II: on-demand, and type III: emergency request. Each of these requests will be associated with different QoI values on the delivered data, as desired by the user. We identify latency (L), reliability (R), and throughput (T) as the three attributes, whose combined value will decide the QoI associated with the delivered data. Energy efficiency is another important parameter that affects the network's performance and impacts the network lifetime, and we will considered it while making decisions in the network, especially when related with choosing a data delivery path. We will not use absolute values of these attributes in deciding the QoI value. Instead, we will associate priorities with each of these attributes for every request type, and let these priorities decide the importance of the absolute value of the attributes. Thus, each request type is classified according to the priorities associated with QoI attributes of L, R, and T, and the importance of considering energy efficiency in making a decision choice in the network, as shown in Table 3.

The QoI attributes are monitored from feedback in the network. When a data packet is transmitted from a CN to its one-hop neighbors, the QoI attribute values are piggy-backed along with the acknowledgement it receives from these nodes. These values will be stored in a knowledge base (KB) and used in deciding the most appropriate next hop for sub-sequent requests arriving at that CN. This way, decisions about data delivery paths are dynamic and always based on both the user requirements and the network conditions at any point. The advantage of making these decisions at CNs is that it helps in decentralized decision making. Moreover, only local, one-hop neighborhood information needs to be monitored and stored. This means that the size of the KB to be maintained remains reasonable and can be easy to update and maintain.

Table 3 Priority of QoI attributes for different request types

		QoI attributes		
Request type	Latency	Reliability	Energy	Throughput
Type I: periodic	x	3	1	2
Type II: on-demand	1	2	4	3
Type III: emergency	1	1	x	2

Table 3, shows the association of QoI attribute priorities with each request type. The numbers in the table indicate the priority associated with the attribute. Number 1 indicates top priority and number 3 indicates least priority. The "x" in Table 3 indicates a "don't care" condition. This means that there are no strict requirements on the value of the QoI attribute marked with an "x," and its value does not impact the decision making.

Next, we look at the structure of the KB, where all the information gathered from observations in the network, and learnt from feedback is stored. The KB thus stores all the relevant and useful information that the learning and cognitive decision making algorithms can use. It also serves as a case base which the learning heuristic can use to map a given problem with and decides on the best course of action for the future. Table 4 represents the KB at a cognitive node and has a sample of the information stored in it as attribute-value pairs. Attribute-value pairs are one of the techniques used for knowledge representation. Information is represented in a way that the user can derive useful information from it, by drawing inferences about how the values are connected [54]. The inferences drawn could be based on rules, or heuristics based on learning from observations and feedback in the network. In Table 4, a recursive representation of attribute-value pairs has been used. That is each entry in the *value* field can be another attribute-value pair. What makes this representation effective beyond the attribute-value association is that information can be derived by reading the values along the column too, except for the field containing sensor data. For example, in the *Attribute'* field "Node type" is associated with three values: "Sink," "RN," or "CN." RN represents relay nodes that

Table 4 Sample of a knowledge base and its contents at the cognitive node

Attribute	Value			
1-hop neighbors	Attribute'	Value'		
	Node type	Sink	RN	CN
	Distance	400 m	250 m	350 m
	Remaining battery	∞	200	300
QoI	Request type	I	II	III
	Node	RN3	RN6	SINK
	QoI attributes	$L = \alpha_1$ $R = \beta_1$ $T = \gamma_1$	$L = \alpha_2$ $R = \beta_2$ $T = \gamma_2$	$L = \alpha_3$ $R = \beta_3$ $T = \gamma_3$
Sensor data	Temperature	25		
	Humidity	20		
	UV index	5		
	Carbon monoxide	250		

the CN is connected with, and CN represents other cognitive nodes that the given CN is connected with in the ICSN. The "Distance" field corresponds with the values in the "Node type" and represents the separation between the CN housing the KB and the Sink, RN, and CN, respectively. Tracking the remaining battery level at each of the one-hop nodes helps the CN take energy-aware decisions in choosing the data delivery path. The next major attribute we have used is "QoI." It has information about the "Request Type" (as described by Table 3) that the node has served, next hop "Node" that can best serve each request, and the values recorded for each of the "QoI attributes" of latency (L), reliability (R), and throughput (T) during the previous communication. These values could be different between any pairs of nodes and are thus represented by α^*, β_*, and γ_*. It should be noted however, that this table is only a representation of how information can be stored in the KB. In actual implementation, details of the semantics will have to be worked out to make the representations shorter and effective. In the proposed ICSN framework, we even segregate the routing table from the KB to keep routing decisions simple. Routing tables at the CNs store information only about reaching the one-hop neighbors, not the end-to-end paths, as shown in Table 5. These entries are derived from the KB of Table 4. In Table 5, the "Possible next hops" field suggests the best next hop node for cognitive node 2 (CN2) to transmit data, based on the "Request type." It shows that CN2 is directly connected to the Sink, connected to four RNs that are linked with the Sink, and is also connected to two other CNs. CN-CN paths are not preferred and are represented by the hyphens in the "Request type" column. This is due to the high cost in terms of energy consumption, and the possibility of running into loops without reaching the Sink. These tables can be updated every time the learning algorithm identifies better paths for each request type, based on the changing network dynamics as reflected from the KB. In addition, a reasoning algorithm to help the learning agent in making cognitive decisions must be identified. These are still open research issues that need to be addressed in the future.

Table 5 Routing table at the CN

Routing table for CN 2	
Possible next hops	Request type
Sink	III
RN6- > Sink	II
RN7- > Sink	II
RN2- > CN1	I
RN3- > CN3	I
CN1	–
CN3	–

5 Use-case and performance evaluation

In this section, we provide some performance evaluation of the different learning techniques in improving large-scale ICSNs for Cloud- and/or IoT-based applications. As described in the context of cognitive psychology [49], the learning heuristics will be used as strategies that ignore a part of the information to make decisions faster, and sometimes more accurately compared to more complex methods [25]. We utilize an online version of the A* heuristic search algorithm, which learns from the information available in the knowledge base of the cognitive nodes. We call this *learning data delivery A* (LDDA*) algorithm.* The heuristics will be used to make approximate decision choices, as opposed to optimal decision choices. We compare this with a *cumulative-heuristic accelerated learning (CHAL)* technique that accumulates the heuristic values at each state (relay and cognitive nodes) and makes use of as much information as possible from observations made in the network before making the data delivery path choices. It also uses negative heuristic weights to punish poor next hop node choices, such as revisiting a node along a data delivery path. This way, LDDA* and CHAL will differ in the heuristic weights accumulated by the learning process. Since learning is typically used to improve the decisions made by reasoning engine in cognitive networks, we implement LDDA* and CHAL in a network that uses an analytic hierarchy process (AHP)-based reasoning technique at the cognitive nodes to make data delivery decisions. (The details of AHP-based data delivery (AHPDD) have been described in our previous work [21]). Performance of the heuristically accelerated learning techniques LDDA* and CHAL are compared against the non-learning AHPDD in terms of the QoI observed at the Sink where data is delivered at the end of each transmission round. The algorithms will also be compared in terms of the rate of successful data delivery and the energy consumed during the data delivery process at the end of network's lifetime. The knowledge of the deterministic deployment of the RNs and CNs, and the knowledge accumulated in the knowledge base (KB) of the cognitive nodes, will be used to update the weight of the heuristics during network operation. We evaluate and compare the performance of the aforementioned algorithms using MATLAB simulations. In the following, a brief description of the simulation setup and targeted metrics/parameters in evaluating the performance of the algorithms. Simulations results and a detailed analysis of the results are also presented in this section.

5.1 Simulation setup

The network is setup as described in Fig. 7, with randomly deployed sensor nodes and fixed deployment of relay and cognitive nodes. Simulation parameters are as described in Table 6. Energy deductions at the local cognitive nodes (LCNs) and relay nodes (RNs) during data transmission are as represented in Table 7, based on the transmit powers. The transmit power at RNs is fixed at 3 dBm, and it can be adapted at the LCNs to improve the probability of successful transmission as described in [21]. Data delivery paths from source LCNs in the network are initially established based on AHP analysis of paths along next hop-neighboring RNs. Heuristic learning is introduced in this simulation to increase the average success rate of data delivery to the sink, irrespective of the randomness with which the requests for different traffic types are generated in the network.

The following are the three performance evaluation metrics that will be used to compare the performance of the aforementioned three algorithms:

1. Network lifetime: number of transmission rounds till all one-hop nodes to GCN/Sink node are dead (including RNs and LCNs)
2. Success rate (ρ): it is defined as the ratio of the number of successful transmissions *s* to the sink over the total number of transmission rounds *T* during the network's lifetime. This is represented by Eq. (1) as follows:

$$\rho = \frac{s}{T} * 100 \tag{1}$$

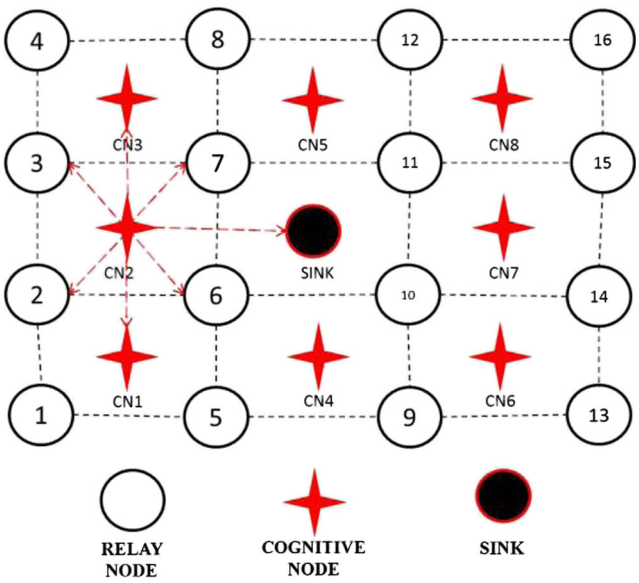

Fig. 7 Use-case scenario setup for simulations

Table 6 Simulation parameters

Parameter	Value
Targeted area	1050 m × 1050 m
Number of nodes	SNs: 1500
	RNs: 16
	LCNs: 8
Transmit power (dB)	SN: <3
	RN: 3
	LCN: {3, 5, 7}
Communication range (m)	SN: 175
	RN: 250
	LCN: 350
	GCN: 500
Application payload size	121 bytes
Per node offered load	0–1400 bits per second

3. Failure rate (ϕ): it is the ratio of the number of failed transmissions *f* to the sink over the total number of transmission rounds T during the network's lifetime. This is represented by Eq. (2) as follows:

$$\varnothing = \frac{f}{T} * 100 \tag{2}$$

4. eQoI: *effective QoI* or eQoI is the heuristics estimate of the QoI associated with data delivered to the sink at the end of successful transmission round. In other words, a heuristic estimate of the value of the QoI at the last hop that delivered the information to the sink.

5.2 Simulation results and analysis

Simulation results for the aforementioned three techniques are summarized in Table 8.

As shown in the above table, results from the simulation using AHP analysis (AHPDD) suggests that during an average lifetime of 78 transmission rounds, the average success rate is 63 %, and the average failure rate is 37 %. However, during the worst case, transmissions can fail for over 50 % of the requests, as suggested by the worst case

Table 7 Transmit power consumption

Ptx (dBm)	Lifecycle reduction (units)
3–5	2
5–7	3
7–9	4
≥10	5
3–5	2
5–7	3

Table 8 Summary of simulation results

Method	Lifetime (rounds)	Average success rate	Average failure rate	Best case success rate	Worst case failure rate
AHPDD	78	63	37	79	52
CHAL	59	84	16	90	19
LDDA*	56	88	12	92	22

failure rate. With the cumulative-heuristic accelerated learning (CHAL), it was found that the average success rate increased to 84 %, and the worst case failure rate was as low as 19 %. The best case success rate was 90 %, which was only 6 % off from the average success rate. This shows that the heuristics performed consistently well under various traffic loads and request arrival patterns. The performance of CHAL was matched very closely by the LDDA* heuristic search algorithm, which provided an 88 % data delivery success rate, but a slightly higher failure rate of 22 % in the worst case scenario when compared with CHAL. Since it is more desirable to have a higher success rate in smart IoT applications, we further compare the performance of LDDA* and CHAL techniques in terms of their effective QoI (eQoI) as observed at the Sink to identify the best heuristic of the two. Figure 8 shows the result of the comparison of the eQoI values for LDDA*, and CHAL, with AHPDD, which does not use any form of learning at the LCNs. In general, we observe that using some form of learning at the LCNs improves the eQoI of the data delivered to the Sink. Of the learning techniques, we observe that LDDA* performs the best in terms of consistently delivering data with higher eQoI at the sink, even towards the end of the network's lifetime. Now, this eQoI is the hop-over-hop value of QoI associated with the data delivered to the sink with respect to latency, reliability, and throughput. Apart from the hop-over-hop latency, the cumulative delay in receiving a response from the network for a request is reflected by the number of hops taken along the path from source to sink. Thus, we can conclude that of the two proposed techniques, LDDA* is capable of delivering data to the sink with a higher average success rate, and better eQoI. Either of these techniques may be used for data delivery in the cognitive ICSN for IoT applications, based on the application and end user requirements (i.e., eQoI), rate of successful data delivery, and cumulative delay from source to sink.

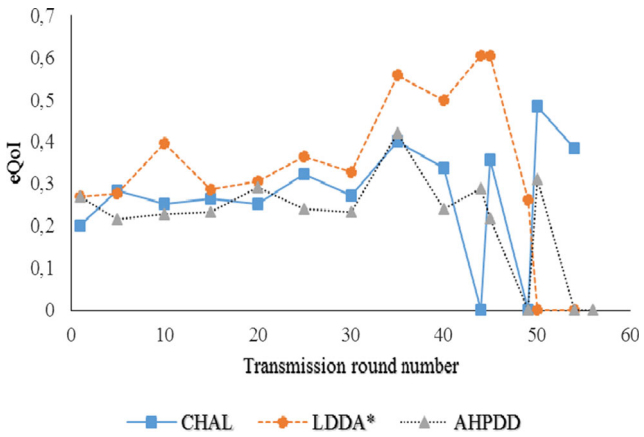

Fig. 8 eQoI as observed at the Sink over the network lifetime

6 Conclusion and open issues

In this paper, we have approached WSNs as information-centric networks, which enable retrieving more information than just the data in attribute-value pairs from the network. We reviewed some routing protocols in DCSNs and presented our views on how learning could have been used to improve their performance. Then, we went on to see the various learning techniques available in the AI domain and found that RL methods are more suitable for use with sensor networks. This is because reinforcement learning involves learning while interacting with the environment which is important in WSN environments where network dynamics change due to changing channel conditions, node deaths, and changing traffic conditions, to name a few. We explored some more RL techniques, and analyzed what the ICSNs should learn, before arriving at a suitable technique for implementing learning in WSNs. Network lifetime and quality of information are the primary features that must be improved by the use of learning algorithms. This work presents a preliminary assessment of the potential advantages of introducing learning in WSNs. A detailed assessment of the best way to do so, and comparison with other techniques with respect to their impact on ICSN lifetime and QoI awareness will provide a more accurate evaluation of the benefits of introducing learning in sensor networks. Thus, we open up the ICSN paradigm as a research area with interesting possibilities. This study opens the door to develop application-/domain-specific ontologies for better knowledge representation at a higher level. The creation of such domain ontologies contributes towards the development of an enterprise architecture framework that can be applied to different application domains using the same underlying cognitive sensor network platform. More functions could be

incorporated at the *sink* to integrate it with the next generation networks constituted of cognitive radio enabled nodes, working in cognitive network setup. The expansion of the sink node functions to a cognitive gateway node can be considered. The cognitive gateway node would then be able to take requests directly from different wireless users such as cell phone users, wireless access points and base stations. Furthermore, the idea of "cognitive elements" shall be used in the intermediate routers of the future internet to provide on-demand content to users quickly.

References

1. Al-Turjman F, Gunay M (2016) CAR approach for the internet of things (IoT). IEEE Can J Electr Comput Eng 39(1):11–18
2. Friend D (2009) Cognitive networks: foundation to applications, Ph.D. Dissertation, Electrical and Comput. Eng., Virginia Polytechnic and State Univ., Blacksburg
3. Bisdikian C, Kaplan L, Srivastava M (2013) On the quality of information in sensor networks, ACM Trans. Sensor Netw, Vol. 9, no. 4, Article 48
4. Pearson DJ, Laird JE (2005) Incremental learning of procedural planning knowledge in challenging environments. Comput Intell 21:4
5. Lazarescu MT (2013) Design of a WSN platform for long-term environmental monitoring for IoT applications. IEEE J Emerg Sel Top Circuits Syst 3(1):45–54
6. (2008) ITU-T technology watch briefing report series, No.4, Ubiquitous Sensor Networks
7. (2010) ITU-T. Series Y recommendations: requirements for support ubiquitous sensor network (USN) applications and services in NGN environment. Y.2221
8. Kalevi K (2008) Quality of experience in communications ecosystem. J Univ Comput Sci 14(5):615–624
9. Hoes R, Basten T, Tham C, Corporaal H (2009) Quality-of-service trade-off analysis for wireless sensor networks. Perform Eval 66(3): 191–208
10. Chen D, Varshney PK (2004) QoS support in wireless sensor networks: a survey. In *International Conference on Wireless Networks*, vol. 233, pp. 1-7
11. Martínez J, Garcí A, Corredor I, López L, Hernández V, Dasilva A (2007) QoS in wireless sensor networks: survey and approach. In *Proc. of the Euro American conference on Telematics and information systems*, p. 20. ACM
12. Bisdikian C, Kaplan LM, Srivastava MB (2013) On the quality and value of information in sensor networks. ACM Trans Sensors Netw 9(4):1–26
13. Tolstikov A, Tham CK, Biswas J (2006) Quality of information assurance using phenomena-aware resource management in sensor networks. In *Networks, 2006. ICON'06. 14th IEEE Int. Conference on*, vol. 1, pp. 1-7. IEEE
14. Ahlgren B, Dannewitz C, Imbrenda C, Kutscher D, Ohlman B (2012) A survey of information-centric networking. IEEE Commun Mag 50(7):26–36
15. Anastasi G, Conti M, Di Francesco M, Passarella A (2009) Energy conservation in wireless sensor networks: a survey. Ad Hoc Netw 7(3):537–568
16. Intanagonwiwat C, Govindan R, Estrin D, Silva F (2003) Directed diffusion for wireless sensor networking. IEEE Trans Netw 11(1):2–16
17. Heinzelman W, Kulik J, Balakrishnan H (1999) Adaptive protocols for information dissemination in wireless sensor networks, Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom_99), Seattle
18. Yu Y, Estrin D, Govindan R (2001) Geographical and energy aware routing: a recursive data dissemination protocol for wireless sensor networks, UCLA Computer Science Dept. Technical Report, UCLA-CSD TR-01-0023
19. (2008) ZigBee Specifications. [Online]. Available: http://www.zigbee.org ZigBee Document 053474r17
20. Al-Fagih A, Al-Turjman F, Hassanein H (2012) Ubiquitous Robust Data Delivery for Integrated RSNs in IoT, in IEEE Global Commun. Conf. (GLOBECOM'12), Anaheim, 3-7 pp. 298-303
21. Singh G, Al-Turjman F (2016) A data delivery framework for cognitive information-centric sensor networks in smart outdoor monitoring. Elsevier Comput Commun 74:38–51
22. Singh GT, Al-Turjman FM (2014) Towards Prolonged Lifetime for Large-scale Information-Centric Sensor Networks, In Proc. of 27th Biennial Symposium on Communications, Kingston, ON, pp. 87-91
23. Singh GT, Al-Turjman (FM) Cognitive Routing for Information-Centric Sensor Networks in Smart Cities, IWCMC, 2014, pp. 1124-1129
24. Eriksson A, Ohlman B, Persson KA (2012) What are the services of an information-centric network, and who provides them?" IEEE AP2PS
25. Al-Turjman F, Al-Fagih A, Hassanein H (2013) A Value-Based Cache Replacement Approach for Information-Centric Networks, *In Proc. of the IEEE Local Computer Networks (LCN)*, Sydney, pp. 902-909
26. Al-Turjman F, Hassanein H (2013) Enhanced data delivery framework for dynamic Information-Centric Networks (ICNs), *In Proc. of the IEEE Local Computer Networks (LCN)*, Sydney, pp. 831-838
27. Al-Fagih A, Al-Turjman F, Alsalih W, Hassanein H (2013) A priced public sensing framework for heterogeneous IoT architectures. IEEE Trans Emerg Top Comput 1(1):133–147
28. Haykin S (2005) Cognitive radio: brain-empowered wireless communications. IEEE J Sel Area Commun 23:201–220
29. Mitola J, Maguire GQ (1999) Cognitive radio: making software radios more personal. IEEE Pers Commun 6(4):13–18
30. Thomas RW, Friend DH, DaSilva LA, MacKenzie AB (2006) Cognitive networks: adaptation and learning to achieve end-to-end performance objectives. IEEE Commun Mag 44(12):51–57
31. Friend DH, Thomas RW, MacKenzie AB, DaSilva LA (2007) Distributed learning and reasoning in cognitive networks: Methods and design decisions, in Cognitive Networks - Towards Self-Aware Networks (Q. H. Mahmoud, ed.), pp. 223-246, John Wiley & Sons
32. Kulkarni RV, Forster A, Venayagamoorthy GK (2011) Computational intelligence in wireless sensor networks: a survey. Commun Surv Tutorials IEEE 13(1):68–96
33. Frster A (2007) Machine Learning Techniques Applied to Wireless Ad-Hoc Networks: Guide and Survey, in *Proc. 3rd Intl. Conf. Intelligent Sensors, Sensor Networks and Information Processing* (ISSNIP)
34. Förster A, Murphy AL (2007) FROMS: Feedback routing for optimizing multiple sinks in WSN with reinforcement learning, in *Proc. 3rd Int. Conf. Intelligent Sensors, Sensor Netw. Inf. Process*
35. Forster A, Murphy AL (2008) Balancing Energy Expenditure in WSNs through Reinforcement Learning: A Study, in Proc. of 1st Int. Wrkshp on Energy in Wireless Sensor Networks
36. Di M, Joo E (2007) A survey of machine learning in wireless sensor networks, *Proc. 6th Int. Conf. Inf., Commun. Signal Process*
37. Sutton RS, Barto AG (1998) Reinforcement learning: an introduction. MIT Press, Cambridge

38. Kaplantzis S, Shilton A, Mani N, Sekercioglu YA (2007) Detecting Selective Forwarding Attacks in Wireless Sensor Networks using Support Vector Machines," *3rd International Conference on*, vol., no., pp.335-340, 3-6
39. Biswas PK (2005) Architecting multi-agent systems with distributed sensor networks," *Integration of Knowledge Intensive Multi-Agent Systems, 2005. International Conference on* , pp.161-166
40. Shenai K, Mukhopadhyay S (2008) Cognitive sensor networks, in *Proc. IEEE 26th Int. Conf. Microelectronics (MIEL)*, pp.315–320.
41. Boonma P, Suzuki J (2008) Exploring self-star properties in cognitive sensor networking, in *Proc. IEEE/SCS Int. Symp. Performance Evaluation Comput. Telecommun. Syst. (SPECTS)*, Edinburgh, pp. 36–43
42. Perera C, Zaslavsky A, Christen P, Georgakopoulos D (2013) Context aware computing for the internet of things: a survey. IEEE Commun Surv Tuts 99:1–41
43. Singh G, Al-Turjman F (2016) Learning data delivery paths in QoI-aware information-centric sensor networks. IEEE Internet Things J 99:1–9
44. Terry WS (2006) Learning and memory: basic principles, processes, and procedures. Pearson Education, Inc., Boston
45. Atkeson CG, Santamaria JC (1997) A comparison of direct and model-based reinforcement learning, *International Conference on Robotics and Automation*
46. Bianchi RAC, Ribeiro CHC, Costa AHR (2007) Heuristic selection of actions in multiagent reinforcement learning. IJCAI'07, Hyderabad
47. Goldstein DG, Gigerenzer G (1996) Reasoning the fast and frugal Way: models of bounded rationality. Psychol Rev 103:650–666
48. Bianchi RAC, Ribeiro CHC, Costa AHR (2012) Heuristically Accelerated Reinforcement Learning: Theoretical and Experimental Results. In *ECAI*, pp. 169-174
49. Gigerenzer G, Gaissmaier W (2011) Heuristic decision making. Annu Rev Psychol 62:451–482
50. Gigerenzer G, Todd PM, The ABC Research Group (1999) Simple heuristics that make us smart. Oxford University Press, New York. ISBN 0-19-514381-7
51. Bianchi RAC, Ros R, De Mantaras RL (2009) "Improving reinforcement learning by using case based heuristics." In *Case-Based Reasoning Research and Development*, pp. 75-89. Springer Berlin Heidelberg
52. Geffner H (2010) Heuristics, Planning and Cognition, In R. Dechter, H. Geffner, and J. Halpern, editors, Heuristics, Probability and Causality. A Tribute to Judea Pearl. College Publications
53. Tversky A, Kahneman D (1974) Judgment under uncertainty: heuristics and biases, new series. Science 185(4157):1124–1131
54. Davis R, Shrobe H, Szolovits P (1993) What is a knowledge representation? AI Mag 14(1):17–33