ISSN: 2458-8989

Natural and Engineering Sciences

NESciences, 2024, 9 (2): 198-212 doi: 10.28978/nesciences.1569561

Improving Communication Networks to Transfer Data in Real Time for Environmental Monitoring and Data Collection

Dr. Liu Zigui [1], Dr. Felicito Caluyo [2*], Dr. Rowell Hernandez [3], Dr. Jeffrey Sarmiento [4], Dr. Cristina Amor Rosales [5]

[1] College of Engineering, Batangas State University the National Engineering University, Alangilan Campus, Batangas City 4200, Philippines. E-mail: 21-04469@g.batstate-u.edu.ph

[2*] College of Engineering, Batangas State University the National Engineering University, Alangilan Campus, Batangas City 4200, Philippines. E-mail: felicito.caluyo@g.batstate-u.edu.ph

[3] College of Engineering, Batangas State University the National Engineering University, Alangilan Campus, Batangas City 4200, Philippines. E-mail: rowell.hernandez@g.batstate-u.edu.ph

[4] College of Engineering, Batangas State University the National Engineering University, Alangilan Campus, Batangas City 4200, Philippines. E-mail: jeffrey.sarmiento@g.batstate-u.edu.ph

[5] College of Engineering, Batangas State University the National Engineering University, Alangilan Campus, Batangas City 4200, Philippines. E-mail: ristinaamor.rosales@g.batstate-u.edu.ph

Abstract

Integrated communication networks (CN) have proven successful in tracking environmental activities, wherein several sensors are installed throughout diverse surroundings to gather data or observe certain events. CNs, comprising several interacting detectors, have proven effective in various applications by transmitting data via diverse transmission methods inside the communication system. The erratic and constantly changing surroundings necessitate conventional CNs to engage in frequent conversations to disseminate the latest data, potentially incurring substantial connection expenses through joint data gathering and dissemination. High-frequency communications are prone to failure due to the extensive distance of data transfer. This research presents a unique methodology for multi-sensor environmental monitoring networks utilizing

*Corresponding Author: Dr. Felicito Caluyo, E-mail: *felicito.caluyo@g.batstate-u.edu.ph*

autonomous systems. The transmission system can mitigate elevated communication costs and Single Point of Failing (SPOF) challenges by employing a decentralized method that facilitates in-network processing. The methodology employs Boolean systems, enabling a straightforward verification process while preserving essential details about the dynamics of the communication system. The methodology further simplifies the data collection process and employs a Reinforcement Learning (RL) technique to forecast future events inside the surroundings by recognizing patterns.

Keywords:

Deep learning, environment, defect detection, convolution neural networks, environment, data transmission.

Article history:

Received: 20/05/2024, Revised: 19/07/2024, Accepted: 23/08/2024, Available online: 30/09/2024

Introduction

Using communication networks (CN) to acquire environmental data has garnered significant interest from computer science academics in recent years (Bai et al., 2020). Environmental surveillance encompasses the procedures necessary to assess the status of the environment regarding air and water quality, soil temperatures, and other variables (Liu et al., 2021). The objectives of employing CNs are categorized into two types: data on the environment acquisition or specific event surveillance. The current and prospective uses in environmental information collecting encompass species in danger conservation, zoological research, pollution assessment, maritime surveillance, etc.

Researchers typically blend many sensor types to acquire diverse information about the environment (Li et al., 2023). Numerous current applications utilize various sensors inside the network to track specific events, including hallway surveillance, intruder identification, forest fire identification, and forecasting floods. In event tracking, scientists gather data that examines environmental data utilizing a single sort of sensor. In CNs, every device is often denoted as a node, capable of communicating with a base station that stores and analyzes all data (Bansal & Kumar, 2020).

The effectiveness of such uses is mainly due to the sensor network's communications capability for tracking the extensive environment, given the inherent difficulties of the activities involved (Escher et al., 2020). Observing the intricate and dynamic large-scale surroundings necessitates a considerable number of detectors. CN can efficiently produce extensive and multidimensional information these sensors gather via continual environmental surveillance tasks. Unstructured environmental information poses significant challenges for analysis and pattern recognition (Liu et al., 2022). The CN has issues with the transmission of information failures.

The malfunction of the central station might immediately fail the ecological surveillance mission, rendering the remainder of the system ineffective. As the quantity of CN nodes in the system significantly rises, the communication volume inside the network will similarly escalate (Rao et al., 2023). This study proposes the development of CNs for the ecosystem and the environment, utilizing concepts from dynamical systems, wherein each node gathers and stores data. The research uses a Boolean system, a specific dynamical structure, to represent information flow and aggregation throughout the monitoring task, employing a straightforward Boolean rule (Jiménez-Hernández et al., 2020). The study specifically focuses on modeling the interactions among the instruments and examines the dynamics generated by these interactions throughout an environmental surveillance assignment.

The research experiments in an artificial setting to examine the scalability of coverage of areas and resilience to network failures. The experiment's findings demonstrate that the approach can address these constraints, mainly using Reinforcement Learning (RL) techniques at every node. The research performs simulations to prove that the sensors can achieve consensus on an event. The concept is advantageous when utilizing error-prone detectors, which are prevalent and cost-effective, making them appealing for extensive CNs.

Real-time Data Transmission Algorithm

Q-Learning (QL) algorithm is an off-policy TD control algorithm, the most widely used algorithm among several RL methods, and an essential breakthrough in RL (Maoudj & Hentout, 2020).

- ***System Model***

In CNs, a network with a known topology comprises M sensor nodes and one base station (BS), where the BS is the data receiving center, and the rest are sensor nodes (Chang et al., 2020). These nodes periodically generate data with different deadlines, which are transmitted between multiple nodes and sent to the BS in time slots.

The Q-learning model is adopted to determine the transmission nodes for each time slot for data transmission scheduling. This model consists of four elements: state space S (set of all data states), action space A (set of transmitted actions), learning strategy (time slot allocation strategy), and reward R (immediate feedback on action execution effectiveness).

The Deep Q-network (DQN) model achieves supervised learning through a dual Q-network architecture and experience pool technology (Turgut & Bozdag et al., 2020). Two networks have the same structure but different parameters, one for real-time Q-value prediction and the other for stable target Q-value updates to reduce correlation and improve algorithm stability. The experience pool collects state transition samples, trains the network using stochastic gradient descent (SGD), and reduces the correlation between data by covering old data.

- ***Optimal Action Selection Strategy***

Deep Neural Networks (DNN) achieve compact representations of complex functions through multiple hidden layers, where each layer undergoes nonlinear transformations. A stacked autoencoder (SAE) DNN comprises multiple layers of sparse autoencoder network, using unsupervised pre-training combined with a supervised fine-tuning training method (Tang et al., 2020). In the unsupervised stage, the researcher learns hidden features layer by layer. During the fine-tuning phase, the network is optimized based on pre-trained parameters.

This algorithm utilizes a multi-layer SAE model to map system state to behavior and quickly obtain optimal decisions. The input layer of the model contains the status information (node data status, remaining hops, deadline), and the output layer evaluates the urgency of each node data and determines the transmission order. The number of hidden layer neurons is related to the number of nodes, and nonlinear activation functions (such as Recurrent Learning Unit (ReLU), sigmoid, and tanh) are used to process the input.

- ***State-Behavior Mapping Network***

DNN achieves compact representations of complex functions through multiple hidden layers, where each layer undergoes nonlinear transformations. A SAE-DNN comprises multiple layers of sparse autoencoder network, using unsupervised pre-training combined with a supervised fine-tuning training method (Prottasha et al., 2022). In the unsupervised stage, the researcher learns hidden features layer by layer. During the fine-tuning phase, the network is optimized based on pre-trained parameters.

This algorithm utilizes a multi-layer SAE model to map system state to behavior, quickly obtaining the optimal decision. The model structure is shown in Figure 1. The input layer of the model contains the status information (node data status, remaining hops, deadline), and the output layer evaluates the urgency of each node data and determines the transmission order. The number of hidden layer neurons is related to the number of nodes, and nonlinear activation functions (such as ReLU, sigmoid, tanh) are used to process the input.

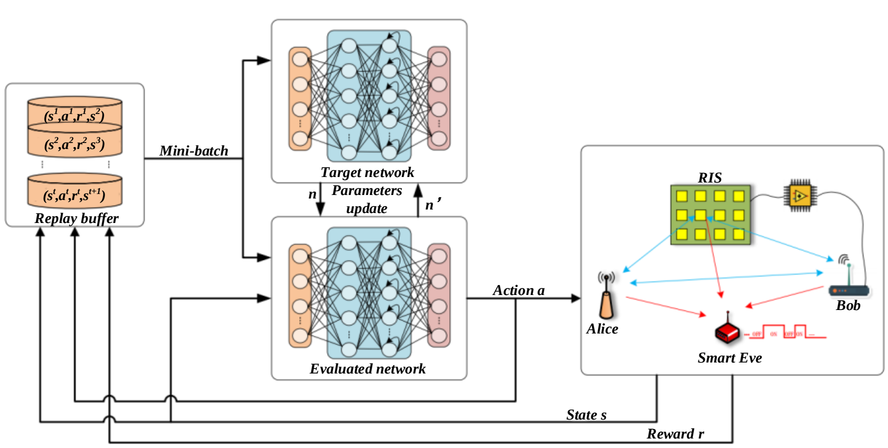

Figure 1. SAE model structure

- ***Fusion of DQN Algorithm and Q-Learning***

The DQN algorithm is designed for scenarios considering communication constraints, interference, remaining deadlines, and hop count changes. It utilizes DNNs to evaluate state action mapping and optimizes it through Q-learning and experience replay (Zhao et al., 2020). The algorithm first constructs a communication interference model to determine the concurrent node set. The DQN scheduling algorithm flow is shown in Figure 2. In the initial stage, Q learns to collect state transition information and store it in the experience pool without training the SAE network. After completing the experience pool, combine DQN supervision to train SAE and optimize data transmission scheduling to reduce packet loss. The algorithm process includes experience pool accumulation, SAE network training, state action recommendation, and Q-value update until the training objectives are achieved. Finally, the trained SAE network will be utilized for data transmission scheduling.

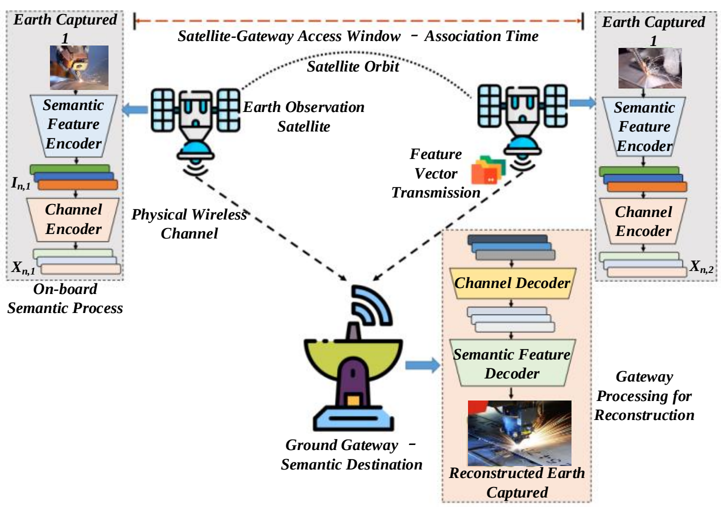

Figure 2. DQN scheduling algorithm

A real-time data transmission scheduling technique utilizing DQN is suggested to tackle the challenge of simultaneous transmission of several data streams. Through the examination of development plans, the most critical data is identified as the current time slot action a, and data suitable for parallel transmission is chosen depending on a and system state. Add a to the waiting queue and continue selecting conflict-free data based on urgency until the queue is full. Action strategies guide this process and establish state and action mappings through the SAE network to optimize data transmission scheduling.

Related Works

The prevalent use of CN is observation and event tracking. Tomas et al. suggest building a CN to address the rise in illegal immigration (Thomas, 2021). Cruz-Paredes et al. built a single CN for tracking threatened fish, utilizing wireless sensors deployed in the water to measure dissolved oxygen, water temperatures, ambient temperature, moisture, and illumination (Cruz-Paredes et al., 2021). Hing et al. established a CN for tracking pre-swarming colony activities (Hong et al., 2020). They devised a prediction algorithm utilizing the recognition of patterns through clustering data mining methods on the repetitive daily actions of bees. Elevated temperatures, food scarcity, and fluctuations in temperature and moisture can induce swarming actions among bees, often resulting in economic losses for farmers. Their technology encounters issues related to data transfer expenses. The expenses are escalating tremendously while attempting to cover more extensive areas.

The CN has been utilized to track human living surroundings. They Established a sensor network to observe activity in the corridor, with 180 load detectors linked to 30 CN nodes. The load detectors are integrated into floor tiles, enabling them to communicate information. Each sensor node is connected to a computer tasked with gathering and accumulating environmental information from the sensor nodes; the resultant information is presented to represent the current condition of the corridor. Gawre, (2022) establish a centralized CN installed at a facility to monitor possible issues such as interrupters, converters, and transformer bearings (Gawre, 2022). Nodes inside the system can interact via a CN that employs a dynamic link-quality routing technique. The collection of sensors relays all sensor information to a base unit that processes and displays data. The advancement of CN has been utilized to monitor environmental noise. Kane et al., (2022) developed a system using a CN founded on the Tmote Invent prototype platform (Kane et al., 2022). They created a system utilizing tiny LAB, a Matlab-based program that facilitates real-time gathering, processing, and visualization of data gathered by the CN. Each of the sensors in the framework supplies sufficient empirical data on noise sources, and the system can evaluate the pollution level utilizing a predetermined noise indicator, which is a calculation.

The server must possess computing capability. The whole network possesses a significant likelihood of a Single Point of Failure. A wireless detection unit was created that exhibits identical capability. The wireless detection unit employs integrated processing techniques for data to assess noise by calculating comparable continuous noise levels, hence reducing data transfer and enhancing the node's overall lifespan (Adil et al., 2020).

In recent years, artificial intelligence has been utilized in environmental surveillance to manage substantial volumes of data. This article employs an RL methodology within a CN. Every sensor inside the network functions as an agent capable of interacting with its surroundings and progressively utilizing feedback to enhance its behavior. In a system that necessitates sensors to make decisions in a highly changing world. RL possesses unique benefits when planning.

Data Analysis and Transmission Management Architecture and System

- ***Heterogeneous Platform for Data Analysis and Transmission Management***

Figure 3 shows that the real-time data analysis and transmission management architecture is built on a high-performance Field Programmable Gate Array (FPGA) and embedded Central Processing Unit (CPU) platform.

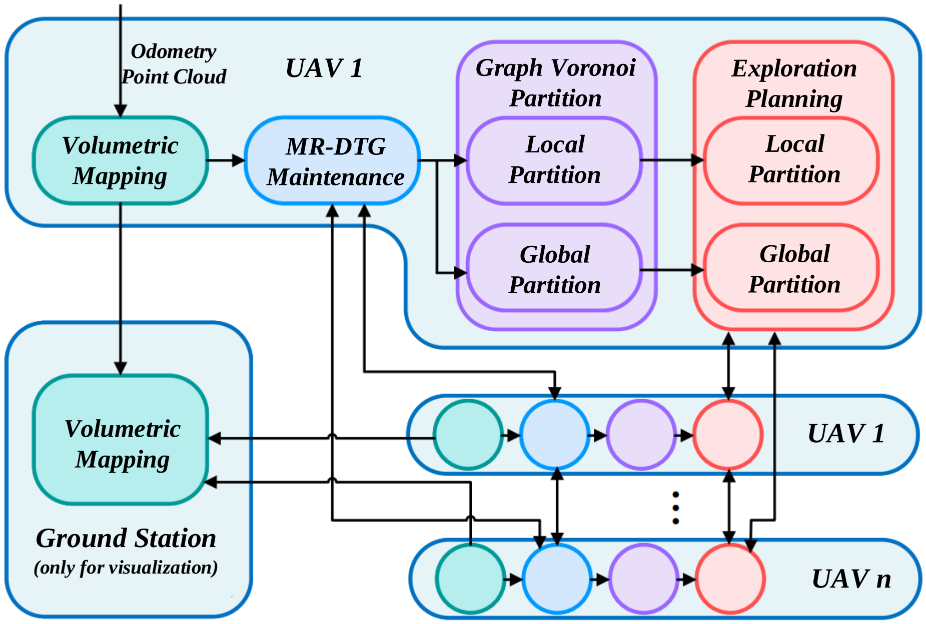

Figure 3. Transmission management architecture

FPGA is the leading data transmission platform. On the one hand, FPGA itself contains rich I/O resources. With Aurora protocol and embedded high-speed serial transceiver, it can flexibly adapt to the data transmission needs of front-end modules and complete high-speed real-time data transmission. Essential functions such as transmission and multi-channel data management and distribution. On the other hand, users can complete the parsing tasks of various custom and general protocols through the wiring and logic resources integrated within FPGA. At the same time, based on the reconfigurable characteristics of FPGA, its custom logic can be programmed according to the needs of actual application scenarios. Complete fundamental digital signal processing functions such as algorithm preprocessing and flexibly complete task indicators for different tasks under conditions such as redefining part of logic or rewriting code. At the same time, the Digital Signal Processing (DSP) softcore integrated into the current high-performance FPGA can realize the complex algorithm logic that is difficult to recognize by the FPGA itself, and the combination of the two can complete the high-speed real-time data algorithm function that is difficult to handle by a single DSP chip. Finally, the Parallel Communication Interface (PCI) hardcore resources integrated into FPGA can be flexibly interconnected with other embedded cores and expansion modules to interact with data and control information, helping FPGA realize more complex functions. Embedded CPU is mainly used as the control core in the whole architecture, and it has multiple dual-threaded processing cores, which can help FPGA decompose control tasks, issue them as commands, and complete the interactive response of the whole task with the interrupt information returned by FPGA. At the same time, FPGA can transfer some complex algorithms and data distribution tasks to an embedded CPU and use the computing power of the CPU to complete complex algorithm processing. Correspondingly, the embedded CPU can unload its complex tasks, such as the network protocol stack, into the FPGA to accelerate the operation and reduce the load and overhead of the CPU.

Moreover, the network protocol module integrated into the CPU can increase the architecture's flexibility and scalability. To sum up, the whole heterogeneous architecture can fully play the advantages of FPGA and embedded CPU and complement each other to form 1 +1 greater than 2 data transmission and processing capabilities, which has a broad application prospect in digital signal processing and transmission. Based on this architecture, developers can quickly build a set of embedded high-speed digital signal transmission and processing systems, which can complete various functions such as digital signal acquisition, high-speed data transmission, multi-channel signal online management, digital signal processing algorithm implementation, and protocol stack offloading.

- ***Overall System Framework***

The whole system framework is formed by extending the heterogeneous processing architecture and consists of three processing boards in total. The first processing board has a high-speed analog-to-digital conversion card to collect the target signal. The signal is processed by the preprocessing algorithm module for channelization, and other algorithms are then transmitted to the Aurora TX module. Finally, it is transmitted to the backboard through the Aurora link, which is called the first processing board. The processing board is an acquisition processing board. The second processing board has an embedded CPU and a high-performance FPGA to form a heterogeneous architecture. The data received by the Aurora module is parsed, managed, and distributed on the FPGA, and the data is uploaded or stored according to the commands issued by the embedded CPU. Called a data processing board, The third processing board is equipped with 1-4 large-capacity SSDs, which receive data from the PCI link and store it according to the file system configured by the software. It is called a storage processing board.

In the system implementation process, other students in the laboratory complete the design of the data acquisition daughter card, algorithm preprocessing module, storage processing board, and embedded CPU software. This paper only involves the construction of the Aurora board-level transmission link, the heterogeneous architecture of the data processing board, the FPGA hardware logic of the data processing board, and heterogeneous architecture software and hardware interaction module design.

Experimental Results and Analysis

Simulation Parameter Configuration

Simulation experiments are conducted on random deadline data packets to evaluate the performance and the algorithms in this chapter in reducing packet loss within the super cycle (i.e., data packets with insufficient remaining deadline to complete the remaining hops). The experiment sets a long time slot (such as 1000T) to compare the packet loss situation of the algorithm within a given time, and other parameters are shown in Table 1.

Table 1. Simulation parameters

Parameter	Value Description
Learning rate	a = 0.02
discount factor	h = 0.8
instant reward r coefficient	k1 = k2 = 0.6
	B = 1 or B = 1.3
Delay reward R coefficient	p1 = 0.1, p2 = 0.2, p3 = 0.3

The experiment uses Load Runner to simulate the sending end of the data stream and installs Load Runner 9.0 on four machines, the data preprocessing program on one machine, and the storage handler program on one machine. The machine is dual-core with 4GB of memory and a 2.4 GHz CPU. Install HBase-1. 3.1 on four machines, cooperate with Hadoop-2. 7.4, one of which plays the role of the controller node, and the data is finally stored in HBase. The operating system is Ubuntu 12.04, and the version is JDK 1.8. Table 2 shows node information.

Table 2. Cluster node information

IP address	Host name	dentification	instructions
132.53.121.156	test001	NameNode	Master
132.53.121.13	test002	DataNode	RegionServer
132.53.121.24	test003	DataNode	RegionServer
1132.53.121.122	test004	DataNode	RegionServer

Algorithm Analysis

In the simulation, the real-time data transmission scheduling algorithm based on DQN is called DQN, which is compared with the classical Edge Detection and Forecasting (EDF) algorithm and the enhanced Encoded Data Protection (EDP) algorithm. EDF prioritizes data transmission with the shortest deadline, while EDP divides priorities based on data urgency and remaining time/hop count. This chapter analyzes the packet loss situation of DQN and the other two algorithms under different deadlines and node numbers and evaluates network performance.

Table 3. Average number of packets lost by different algorithms

Algorithm	EDF	EDP	DQN
Number of packets lost	1506	1178	898
Number of packets sent successfully	1598	1927	2207
loss rate	0.465792	0.364224	0.277536

Table 3 presents the average quantity of lost packets across several techniques, with the node data generation interval randomly established between 1.5 and 3.5 times the total hop count of the node. The DQN method outperforms the EDP and EDF methods, with the latter exhibiting the poorest scheduling efficiency, almost half that of DQN. The DQN simulation results are averaged ten times to ensure consistency in comparison.

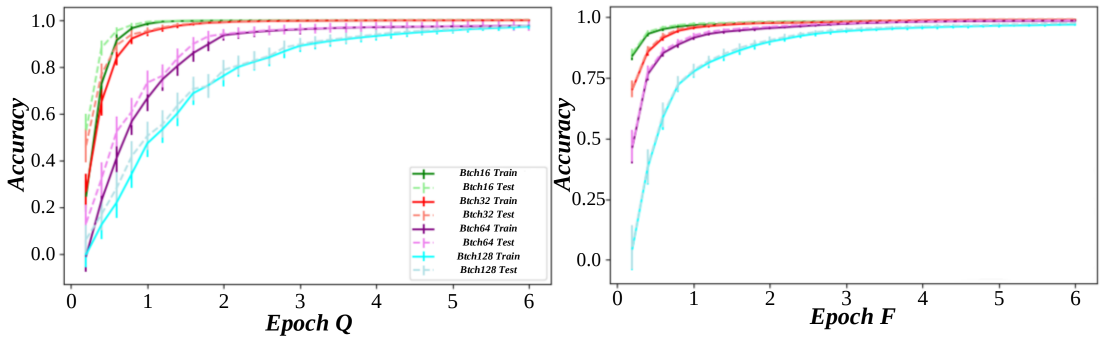

Figure 4. Change of data generation cycle

Figure 4 illustrates the performance variations of three distinct methods (DQN, EDP, EDF) regarding packet loss as the number of sensor nodes incrementally rises from 5 to 25 in a computerized context, with the data subsequent generations cycle established at threefold the total number of hops from sensors nodes to the final nodes. The graph illustrates that as the total amount of nodes rises, the efficiency of the EDP method progressively converges to that of the DQN method. While the EDP method exhibits benefits when the data generating cycle is a considerable integer double the total amount of data transfer hops (three times or more), the general network efficiency of the DQN method remains better. The packet loss rate of the EDF method remains elevated since the system's shortcomings are markedly exacerbated when all data production cycles are a multiple of the total amount of hops, leading to suboptimal efficiency in this context. When the data production cycle is thrice the total number of hops, the DQN and EDP algorithms demonstrate improved efficiency relative to the EDF method.

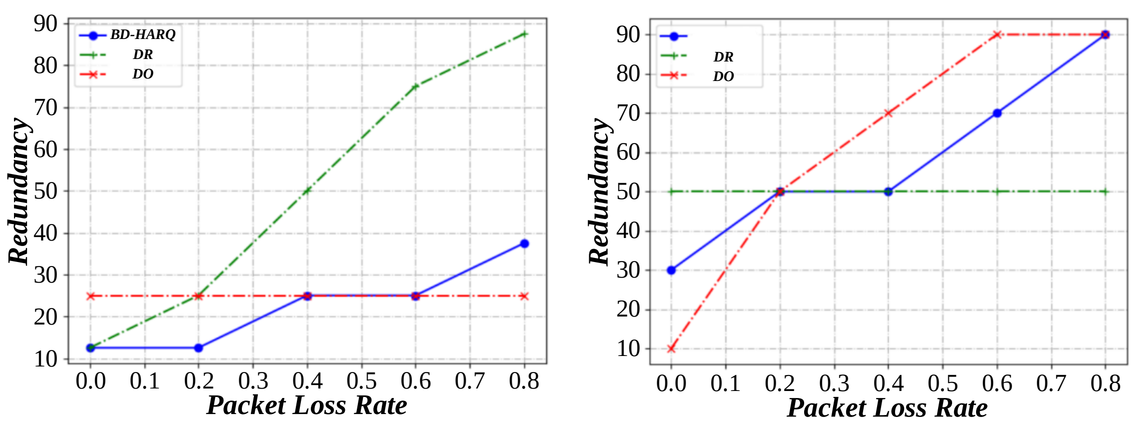

Figure 5. Variation of packet loss number

Figure 5 shows that the data generation cycle is randomly 1.5 to 3.5 times the total number of nodes, and the number of packet losses for the three algorithms changes as the number of nodes increases (5 to 25). The number of packet losses in EDF increases almost linearly, while EDP is not stable enough. After the number of nodes exceeds 20, packet losses surge sharply, indicating poor performance in multi-node random deadline scenarios. DQN performs the most stably, with the least number of packet losses and a steady increase.

Table 4. Link test results

Test Type	Rate test	Functional testing
Rate (Gpbs)	74.5104	70.52724
Efficiency (%)	110.352	/
Error rate (%)	0	0

Table 4 shows the test results of the rate test and function verification, which shows that the link built by Aurora protocol can efficiently and accurately complete board-level real-time data transmission and, at the same time, verify the logical correctness of the encapsulation and de-encapsulation module. The results of the two tests show that the system's Aurora board-level transmission link design can ensure the correct transmission of real-time data with a total bandwidth of 61.866 Gbps between processing boards.

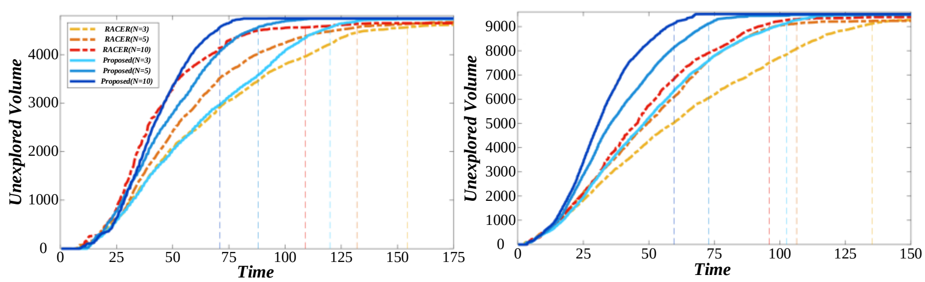

Figure 6. Test delay

A query test was designed to evaluate the optimization effect of query performance, which analyzed the trend by storing and querying data over time. In the experiment, the latency performance of the original HBase system and the system introduced in this chapter were compared when processing these queries. The test covers ten sets of test data ranging from 100000 to 1 million, all of which are arranged in chronological order. The experimental results are shown in Figure 6. As the size of the test data gradually increases, the query latency of both systems shows an upward trend. However, the query latency of the native HBase system has increased dramatically, almost reaching eight times that of the system introduced in this chapter. In contrast, the system proposed in this study performs well in processing these queries, with query latency consistently maintained below 100 milliseconds, demonstrating excellent performance optimization results.

Table 5. System performance test results

Number of drones	throughput	Storage latency	Processing delay	Inquiry time
550	0.385	112.86	107.91	85.25
825	0.572	121.44	131.23	87.34
1100	1.276	138.38	143.44	89.65
1375	1.342	154.66	187.99	91.41
1650	1.287	177.21	221.54	90.53

The system's performance is tested when different numbers of nodes transmit data concurrently. The test was conducted multiple times, and the average value was taken. The system's throughput, storage delay, processing delay, and query time were recorded when the data size corresponded to 500, 750, 1000, 1250, and 1500 nodes, respectively. The performance test results are shown in Table 5 below. According to the above table, the system's massive data support capability, real-time performance, and high concurrent throughput capability have all met the performance index requirements in the performance requirements analysis.

Table 6. Real-time comparison table of simulation process after real-time optimization

Satellite navigation system	GPS	Galileo	GLONASS	Beidou
t (μs)	2688	2433	2526	2905
t (μs) after real-time optimization	2030	1912	1962	2049
P (us)	228241	412237	380190	315134
P (μs) after real-time optimization	51022	37337	30737	40120

It can be seen from Table 6 that after optimizing the information flow processing and transmission process of the data simulation software, the time delay has been reduced, and the time delay of the information flow processing and transmission process of each navigation system has been reduced to less than 1900 microseconds, reaching the requirement of less than 2ms. The variance is reduced, and the time delay fluctuation is slight, which meets the real-time requirements of the data simulation software.

Conclusion

This study presents a multi-sensor environmental tracking system built around a Boolean network. Monitoring an environment generates intricate data that necessitates a system with substantial computational capacity and practical algorithms for processing. The system integrates a Boolean operating exclusively in affirmative or hostile states, 0 or 1. A Boolean network significantly decreases the level of detail of environmental data while preserving essential information. The research utilizes Boolean networks to diminish the detail of data related to the environment while retaining important information. In-grid computation employs an approach to computing that significantly conserves system assets. The method employs RL, which dynamically enhances its actions by administering rewards or punishments depending on average consolidated values.

When the system exhibits inadequate performance in event detection, it incurs a penalty, prompting an instantaneous adjustment of the significance levels in the decision-making element. Upon the system's precise detection of the occurrence, it is awarded a reward, which is then added to the overall reward accumulated by the system. The system first exhibits poor results in event detection; nevertheless, it progressively enhances its capabilities and achieves accurate event monitoring throughout the entire network. The simulation's findings illustrate that the system integrates Boolean networks with reinforcement teaching to yield precise outcomes. The network can precisely identify the moving item by augmenting the weight on the grid aggregating value while reducing the weights on the present sensor input and the past grid aggregating value. The results indicate that prioritizing the grid mean aggregation value is the optimal model for the framework in two of the evaluated cases. In the future, the research wants to conduct trials in grassland or urban environments for diverse ecological tracking tasks.

Author Contributions

All Authors contributed equally.

Conflict of Interest

The authors declared that no conflict of interest.

References

Adil, M., Khan, R., Almaiah, M. A., Binsawad, M., Ali, J., Al Saaidah, A., & Ta, Q. T. H. (2020). An efficient load balancing scheme of energy gauge nodes to maximize the lifespan of constraint oriented networks. *IEEE Access*, 8, 148510-148527.

Bai, L., Zhu, L., Liu, J., Choi, J., & Zhang, W. (2020). Physical layer authentication in wireless communication networks: A survey. *Journal of Communications and Information Networks*, 5(3), 237-264.

Bansal, S., & Kumar, D. (2020). IoT ecosystem: A survey on devices, gateways, operating systems, middleware, and communication. *International Journal of Wireless Information Networks*, 27(3), 340-364.

Chang, K. C., Chu, K. C., Wang, H. C., Lin, Y. C., & Pan, J. S. (2020). Energy saving technology of 5G base station based on internet of things collaborative control. *IEEE Access*, 8, 32935-32946. https://doi.org/10.1109/ACCESS.2020.2973648

Cruz-Paredes, C., Tájmel, D., & Rousk, J. (2021). Can moisture affect temperature dependences of microbial growth and respiration?. *Soil Biology and Biochemistry*, 156, 108223. https://doi.org/10.1016/j.soilbio.2021.108223

Escher, B. I., Stapleton, H. M., & Schymanski, E. L. (2020). Tracking complex mixtures of chemicals in our changing environment. *Science*, 367(6476), 388-392.

Gawre, S. K. (2022). Advanced fault diagnosis and condition monitoring schemes for solar PV systems. *In Planning of Hybrid Renewable Energy Systems, Electric Vehicles and Microgrid: Modeling, Control and Optimization*, 27-59. Singapore: Springer Nature Singapore.

Hong, W., Xu, B., Chi, X., Cui, X., Yan, Y., & Li, T. (2020). Long-term and extensive monitoring for bee colonies based on the Internet of Things. *IEEE Internet of Things Journal*, 7(8), 7148-7155.

Jiménez-Hernández, E. M., Oktaba, H., Díaz-Barriga, F., & Piattini, M. (2020). Using web-based gamified software to learn Boolean algebra simplification in a blended learning setting. *Computer Applications in Engineering Education*, 28(6), 1591-1611.

Kane, M. B., Peckens, C., & Lynch, J. P. (2022). Introduction to wireless sensor networks for monitoring applications: principles, design, and selection. *In Sensor Technologies for Civil Infrastructures*, 335-368. Woodhead Publishing.

Li, Y., Yang, G., Su, Z., Li, S., & Wang, Y. (2023). Human activity recognition based on multienvironment sensor data. *Information Fusion*, 91, 47-63.

Liu, H., Kong, F., Yin, H., Middel, A., Zheng, X., Huang, J., & Wen, Z. (2021). Impacts of green roofs on water, temperature, and air quality: A bibliometric review. *Building and Environment*, 196, 107794. https://doi.org/10.1016/j.buildenv.2021.107794

Liu, X., Lu, D., Zhang, A., Liu, Q., & Jiang, G. (2022). Data-driven machine learning in environmental pollution: gains and problems. *Environmental science & technology*, 56(4), 2124-2133.

Maoudj, A., & Hentout, A. (2020). Optimal path planning approach based on Q-learning algorithm for mobile robots. *Applied Soft Computing*, 97, 106796. https://doi.org/10.1016/j.asoc.2020.106796

Prottasha, N. J., Sami, A. A., Kowsher, M., Murad, S. A., Bairagi, A. K., Masud, M., & Baz, M. (2022). Transfer learning for sentiment analysis using BERT based supervised fine-tuning. *Sensors*, 22(11), 4157. https://doi.org/10.3390/s22114157

Rao, S. P., Chen, H. Y., & Aura, T. (2023). Threat modeling framework for mobile communication systems. *Computers & Security*, 125, 103047. https://doi.org/10.1016/j.cose.2022.103047

Tang, C., Luktarhan, N., & Zhao, Y. (2020). SAAE-DNN: Deep learning method on intrusion detection. *Symmetry*, 12(10), 1695. https://doi.org/10.3390/sym12101695

Thomas, N. (2020). Immigration: The “illegal alien” problem. *International Journal of Group Psychotherapy*, 70(2), 270-292.

Turgut, Y., & Bozdag, C. E. (2020). Deep Q-network model for dynamic job shop scheduling problem based on discrete event simulation. *In IEEE Winter Simulation Conference (WSC)*, 1551-1559.

Zhao, M., Lu, H., Yang, S., & Guo, F. (2020). The experience-memory Q-learning algorithm for robot path planning in unknown environment. *IEEE Access*, 8, 47824-47844. https://doi.org/10.1109/ACCESS.2020.2978077