# Wireless Sensor Networks: Dynamical routing adaptation via Multi-Armed Bandits (MABs)

Emanuele Mengoli*

March 2024

## Contents

**1 Introduction** **2**

**2 State of Art** **2**

**3 Methodology** **3**
3.1 Problem Statement . . . . . 3
3.2 Naive Solution . . . . . 4
3.2.1 Achievements . . . . . 5
3.2.2 Shortcomings . . . . . 5
3.2.3 Deadlock Analysis . . . . . 5
3.3 Towards a low-memory approach: UCB-based dynamic routing algorithm . . . . . 7
3.3.1 UCB-Based Dynamic Routing Algorithm . . . . . 7
3.3.2 Achievements . . . . . 8
3.3.3 Shortcomings . . . . . 8
3.3.4 Algorithms implementations . . . . . 9

**4 Experimental Results** **10**
4.1 $\epsilon$-Greedy vs Default Routing . . . . . 11
4.1.1 Exploration versus Exploitation . . . . . 11
4.2 UCB-Based Heuristics . . . . . 11
4.3 Algorithm Performance Comparison . . . . . 12

**5 Conclusion** **13**

*Project course: "Wireless networks: from cellular to connected objects (INF567)", École Polytechnique

**A Appendix** **18**

**Abstract**

This project's paper focuses on single-hop Wireless Sensor Networks (WSNs), in particular how the inherent single-hop constraint can be relaxed to enable multi-hop routing. The underlying objective is extending the current geographic coverage limits of WSNs, imposed by the single-hop to gateway constraint, to larger geographic areas where end-nodes route packets to an available gateway. The proposed approach leverages MABs to build an adaptive dynamic routing mechanism. This contribution constitutes the project work for the class INF567 - "Wireless networks: from cellular to connected objects" at École Polytechnique.

# 1 Introduction

This project delves into the realm of WSNs, with a focused exploration on developing a multi-hop solution tailored for non-time-sensitive networks. The primary goal is to minimize end-to-end latency between sources and gateways through effective routing strategies. Unlike traditional single-hop transmissions, multi-hop routing offers a significant improvement by potentially reducing the transmission time, requiring sensors with enhanced packet forwarding capabilities. This advancement necessitates the establishment of a robust control plane over the data plane infrastructure, ensuring efficient management of packet forwarding processes.

Building upon the foundational research by Professors João Pedro Pedroso, Sonia Vanier, Juan-Antonio Cordero-Fuertes, and Renan Spencer on the max-flow problem in wireless networks, this work addresses the computational challenges observed in their combinatorial optimization approach. Despite the innovative nature of their study, whose findings, though not yet public, have been shared with the author, the approach proved to be computationally intensive for networks modeled as a 5x5 grid. Consequently, this project proposes a novel methodology, leveraging a bandit-based heuristic to model and solve the maximum throughput problem, offering a more scalable and computationally efficient solution.

The proposed solution's efficacy and robustness will be evaluated on grid networks and randomly generated topologies, aiming to validate its performance under a variety of conditions. The rest of this paper is organized as follows: Section 2 discusses the State of the Art. Section 3 details the Methodology. Section 4 presents Experimental Results. Section 5 draws the Conclusion. Finally, Appendix A provides additional material.

# 2 State of Art

Reinforcement Learning (RL) focuses on improving decision making in systems through learning agents. In parallel, the shift from single-hop to multi-hop

network designs represents a significant area of interest within the field, as the information overhead required for packet forwarding decisions is subject to the native constraints of the network. This is particularly evident when considering Low Power Wide Area Networks (LPWANs) in the context of WSNs, where Internet of Things (IoT) applications often employ single-hop technologies such as Sigfox, LoRa and Nb-IoT, for which multi-hop evolution is strictly subject to hardware capabilities and application purpose.

The evolution to multi-hop networking represents an evolving area of research, attracting considerable attention due to its potential in real-world applications [1, 2, 3, 4, 5, 6], such as those observed in Australia [7]. A critical consideration in this context is the need for routing protocols to adhere to the memory and computational limitations inherent to these networks, encompassing aspects like duty cycling, energy efficiency, data transmission rates, and coverage distance. Conventional routing protocols, designed for cellular and wired networks, may require modifications to align with these constraints.

In conjunction with these developments, reinforcement learning-based routing mechanisms for wireless sensor networks have been explored in the literature [8, 9, 10, 11, 12, 13]. Notably, bandit selection strategies offer the advantage of lower computational demands compared to more complex reinforcement learning architectures, making them well-suited for addressing routing adaptation challenges in multi-hop scenarios within the constraints of LPWANs [8, 9, 12, 13, 14, 15].

Building on this foundation, we propose a dynamic adaptation to the general source-routing protocol aimed at optimizing network throughput by minimizing packet forwarding delay. Similar efforts have been made, such as the exploration of energy-aware routing algorithms [15] for WSNs based on Q-learning extensions, and the development of Quality of Service (QoS) supportive routing protocols that utilize Q-learning algorithm, requiring knowledge of neighboring nodes and an acknowledgment system for the decision-reward process [16].

# 3 Methodology

## 3.1 Problem Statement

In this study, we address the scheduling challenges inherent in a WSN with routing capabilities. Specifically, our problem statement revolves around configuring the network to maximize throughput under scheduling problem scenario. We delineate two distinct network topologies to test the efficacy of our proposed algorithm under different stress conditions:

1. **Grid-based Network Configuration:** This scenario employs a simplified, grid-structured network layout where wireless routers are uniformly placed. This configuration serves as a basis for evaluating the algorithm's performance in a controlled environment.

2. **Random Network Topology:** To simulate more complex and unpredictable real-world conditions, we introduce a randomly generated network topology. This scenario is crucial for testing the algorithm's adaptability and resilience under varied and less predictable stress factors.

Our assumptions for both scenarios are as follows:

- The network operates on a single communication channel to streamline the analysis.
- Each router has the capability to transmit and receive data within a fixed communication radius. This parameter is scaled to represent the end-nodes' coverage in the randomly structured topology and to play with the connectivity in the grid.
- We model the demand within the network, per episode, as one unit that needs to be sent by a randomly selected end-node towards the gateway.
- Additionally, we consider that, despite the broadcast nature of packet transmission, only the chosen next-hop retains the packet for the purpose of forwarding. Neighboring nodes that are not selected as the next-hop will simply discard the packet.
- An end-node is capable of sensing whether its neighbors are currently transmitting. Thus, the availability of the next-hop is modeled as a Poisson process, in line with the Aloha protocol, for simplicity.

Important considerations include:

- To mitigate interference, it is imperative that only one device transmits within the reception range of any receiving device at any given time.
- The objective of maximizing network throughput is conceptually aligned with minimizing the total time (makespan) required for all data packets to reach their intended destinations.
- Additionally, for Upper Confidence Bound (UCB) algorithm, we introduce a secondary, decoupled communication channel. This channel is designated for immediate acknowledgments from the gateway, which are then propagated backward toward the source node, allowing decision-making rewarding.

## 3.2 Naive Solution

The initial approach to solving the scheduling problem involves a straightforward solution that computes the shortest path from each end-node towards the gateway. This computation is performed in a centralized manner, after which the determined paths are stored as default routes in the memory of each end-node. To enhance decision-making at each node, we employ a heuristic algorithm based

on the $\epsilon$-greedy strategy, described in pseudocode 1. In this context, for each end-node $i$, given its neighborhood $I_i$ such that $\forall j \in I_i$, $i$ maintains information in a local storage. This dictionary records each neighbor $j$'s identifier $j(\text{id})$ and the cardinality of its default path towards the gateway ($\|\sigma_j\|$), where $\sigma_j$ denotes the default path and $\|\cdot\|$ denotes the cardinality operator. The epsilon-greedy strategy, thus, probabilistically balances the exploration of new paths and the exploitation of the shortest known path towards the gateway, where with probability $1-\epsilon$, the neighbor with the shortest default route towards the gateway is chosen, and with probability $\epsilon$, a random neighbor is selected. See the pseudocode 1 for implementation details. The table 1 provides the notation used.

#### 3.2.1 Achievements

The implementation of the proposed solution introduces significant benefits in terms of routing flexibility and responsiveness. Specifically, it facilitates rapid routing adaptation in response to the unavailability of default next-hops, directly addressing the critical goal of avoiding waiting times in data transmission. This adaptability is crucial for maintaining efficient communication flow within the network, especially in dynamic or unstable environments where route availability can frequently change.

#### 3.2.2 Shortcomings

Despite its advantages, the proposed algorithm has certain limitations that need consideration. A primary concern is the requirement for each node to be aware of the default path length to each of its neighbors. This necessitates a potentially extensive booting phase for initializing the network, leading to increased traffic overhead. Such overhead becomes particularly problematic when adding new nodes to an already deployed network, as it may significantly impact network efficiency. Furthermore, the additional traffic generated by the routing adaptation mechanism may render this approach less suitable for LPWAN environments. LPWANs are characterized by their low data rates and strict duty cycling constraints, which could be adversely affected by the increased signaling and data transmission demands of the proposed solution.

#### 3.2.3 Deadlock Analysis

In the initial, naive formulation, the occurrence of a deadlock within the system required exploration at each next-hop decision. Let $\|\delta_i\|$ represent the path length from end-node $i$ towards the gateway, with $\delta_i$ being the path itself. Consequently, the probability of encountering a deadlock is bounded by $\epsilon^{\|\delta_i\|}$, where $\epsilon \in [0, 1]$. It is important to note that an $\epsilon$-greedy policy is applied in cases where the default next-hop is not available. Therefore, this probability is further bounded as follows:

Assuming $2T$ represents the vulnerable period, and $\lambda$ the data transmission rate, the probability of avoiding packet collision is defined as [17]:

$$\mathbb{P}_{succ} = e^{-2\lambda T} \tag{1}$$

Given a random variable $\beta \sim \mathbb{U}(0,1)$, the probability of making an $\epsilon$-greedy choice is bounded by:

$$1 - \frac{\mathbb{E}(\beta)}{e^{-2\lambda T}} \leq \mathbb{P}_{\epsilon\text{-}greedy} \leq 1 \tag{2}$$

This is because $\mathbb{P}\left(\beta < \mathbb{P}_{succ}\right) \geq 1 - \frac{\mathbb{E}(\beta)}{e^{-2\lambda T}}$, according to the Markov inequality. Thus, as the ratio $\frac{\mathbb{E}(\beta)}{e^{-2\lambda T}} \to 0$, $\mathbb{P}_{\epsilon\text{-}greedy}$ increases. Consequently, the deadlock probability is bounded by:

$$\mathbb{P}_{deadlock} \leq \left(\epsilon \cdot \mathbb{P}_{\epsilon\text{-}greedy}\right)^{\|\delta_i\|} \tag{3}$$

From a topological standpoint, given that the system employs a source routing protocol to avoid loops by appending visited node-ids in the packet header, the topology must ensure that:

$$I_j \setminus \delta_i \neq \emptyset \quad \forall i \in V, \quad \forall j \in I_i \tag{4}$$

Here, $I_i$ denotes the neighborhood set of node $i$. This implies that for any node $i$ and its neighbors $j$ and $j'$ within $I_i$:

$$\begin{aligned} &\delta_j \cap \delta_{j'} = \emptyset \quad \forall i \in V, \quad \exists (j, j') \in I_i, \quad j \neq j' \\ &\text{such that} \\ &\exists w \in I_j (\text{or} \in I'_j), \quad w \notin \delta_i \end{aligned} \tag{5}$$

These constraints are combinatorial in nature, leading to an exponential increase in computational time as the number of nodes grows. For non-time-sensitive networks, this approach may introduce unnecessary overhead, given the nature of the IoT application. An approximate solution involves allowing the system to automatically penalize paths leading to deadlocks, necessitating an initial learning phase for the network to discover optimal paths to the gateway.

This necessity arises when considering only locally available knowledge for decision-making, meaning the next-hop choice is made without any knowledge of the neighbors' path lengths. Thus, each end-node must rely on a local estimator of the utility of using a neighbor as the next hop, requiring acknowledgment as a form of ground truth confirmation. This method can be implemented using a heuristic approach based on the UCB bandit mechanism. The proposed implementation assumes a reward mechanism based on gateway acknowledgment upon receiving a packet, which occurs on a separate channel and backtracks along the path.

## 3.3 Towards a low-memory approach: UCB-based dynamic routing algorithm

The second phase of our project addresses the identified shortcomings by proposing a heuristic approach that significantly reduces the dependency on extensive neighborhood knowledge for decision-making. This strategy aims to empower each end-node to act as a decentralized, independent learner, utilizing only locally available information. Consequently, the end-nodes are envisioned to dynamically make next-hop decisions and, ultimately, derive their paths to the gateway through a process of trial and error.

Central to this approach is the elimination of the need for a priori knowledge of default paths to the gateway. The essential information for each node under this new paradigm includes:

- The set of reachable neighbors and their availability.
- The list of nodes already visited by a packet to filter out the neighborhood and prevent looping.

This methodology inherently minimizes traffic overhead, as it only necessitates a minimal "Hello" booting phase to ascertain the neighborhood set. The approach maintains the assumption of source routing to avert the formation of loops; hence, at each forwarding step, the node-id is appended to the packet header, ensuring traceability of the path taken.

Moreover, as in the previous scenario, the availability of nodes to receive packets is modeled on an Aloha-like Poisson process, enabling a node to sense the readiness of another node for communication. This adaptive mechanism, grounded in the principles of the UCB algorithm, allows for a more flexible and decentralized routing strategy.

### 3.3.1 UCB-Based Dynamic Routing Algorithm

The UCB1 algorithm, as proposed by Auer et al. [18], addresses bandit problems with bounded rewards, typically in the range $[0, 1]$, but allows for extension through the inclusion of a parameter $\alpha$. The selection strategy for the algorithm at time $t$ is given by:

$$A_t = \underset{a \in \mathcal{A}}{\operatorname{argmax}} \left[ \hat{\mu}_a(t-1) + \sqrt{\frac{\alpha \log(t)}{N_a(t-1)}} \right], \tag{6}$$

where $\hat{\mu}_a(t)$ represents the empirical mean reward of action (or arm) $a$ after $t$ rounds, $N_a(t)$ denotes the number of times action $a$ has been selected up to time $t$, and $\mathcal{A}$ is the action (arms) space.

Originally, UCB1 was proposed with $\alpha = 2$, signifying the confidence level in the exploration term. Subsequent analysis refined this choice, indicating that any $\alpha > 1/2$ is permissible, thereby providing a broader scope for tuning the algorithm's sensitivity to exploration [19, 20].

In the context of dynamic routing within a network, the UCB-based algorithm aims to optimize the selection of next hops. By considering each potential next hop as an arm of the bandit problem, the algorithm dynamically evaluates and adjusts routing decisions based on the empirical success rates of different paths. Let us consider a deterministic rewarding mechanism such that on acknowledgement of the gateway the reward is 1, while in the case of deadlock it is 0. See pseudocode 2 for implementation details.

#### 3.3.2 Achievements

The proposed algorithm offers a scalable, distributed learning mechanism that is decentralized and asynchronous, making it particularly suitable for networks where sensitivity to time delays is not critical. It is designed to address the problem of throughput maximization through a process that relies solely on trial and error experiences, supplemented by a feedback mechanism based on acknowledgments and rewards. A notable advantage of this approach is its independence from the need to know the path lengths from neighbors to the gateway, which significantly reduces traffic overhead during the booting phase. However, it is important to acknowledge that achieving optimal decision-making may require a longer training phase, as the system relies on incremental feedback to converge towards the most efficient next-hop selections.

#### 3.3.3 Shortcomings

One of the primary limitations of the classical UCB bandits, as applied within this context, stems from their design for rewards bounded within the $[0, 1]$ range. This constraint poses a significant challenge for the deadlock penalization mechanism. Specifically, the narrow range between positive rewards and penalties limits the algorithm's ability to quickly differentiate between beneficial and detrimental actions. As a result, several rounds of training may be necessary before a substantial deviation is observed, which effectively steers the decision-making process away from paths that could lead to deadlocks.

### 3.3.4 Algorithms implementations

| Symbol | Description |
|---|---|
| $\epsilon$ | Exploration parameter for $\epsilon$-greedy decision making |
| $\lambda$ | Rate parameter for node availability |
| $T$ | Transmission time |
| $i$ | Current node |
| $P$ | Set of default paths to the gateway node |
| $Z$ | Set of nodes already visited by a packet |
| $I_i$ | Neighbors of a given node $i$ |
| $I_i'$ | Available Neighbors, considering node availability and visited nodes ($Z$) |
| $j$ | Next Hop chosen based on the selection strategy |
| $\beta$ | Random variable $\sim \mathcal{U}(0,1)$ used to determine node availability |
| $D$ | Deadlock Flag, indicating if a deadlock situation occurs |

Table 1: Notation table for $\epsilon$-Greedy and UCB-Based Dynamic Next Hop Selection algorithms

**Algorithm 1** $\epsilon$-Greedy - Dynamic Next Hop Selection

```
function EpsilonGreedyNextHop(i, Z)
    Initialize j as None
    j_def ← DefaultNextNode(i)
    if (j_def == gateway node) OR NodeIsAvailable(β, λ, T, j_def) then
        return j_def
    else
        I_i ← GetNeighbors(i, G)
        I_i' ← I_i ∖ Z AND NodeIsAvailable(β, λ, T, j') (∀j' ∈ I_i)
        if I_i' is empty then
            return None
        end if
        if β ≤ (1 − ε) then
            j ← SelectShortestPathToGateway(I_i', P)
        else
            j ← RandomChoice(I_i')
        end if
    end if
    return j
end function
```

The proposed dynamic routing algorithms incorporate several key functions to optimize packet delivery through the network. The `ShortestPathToGateway` function computes the minimum hop path or least cost path from any given node to the gateway, leveraging Dijkstra algorithm. Concurrently, the `GetNeighbors`

**Algorithm 2** UCB - Dynamic Next Hop Selection

**function** UCBDynamicNextHop($i, Z$)
    Initialize $D$ as False
    $I_i \leftarrow$ GetNeighbors$(i, G) \setminus Z$
    **if** $I_i$ is empty **then**
        $D \leftarrow$ True
        **return** None, $D$
    **end if**
    $I_i' \leftarrow$ NodeIsAvailable($\beta$, $\lambda$, $T$,$j'$) $(\forall j' \in I_i)$  ▷ Filter unavailable nodes
    **if** $I_i'$ is empty **then**
        **return** None, $D$
    **else if** gateway node $\in I_i'$ **then**
        **return** gateway node, $D$
    **else**
        $j \leftarrow$ UCBSelection$(I_i')$
        **return** $j, D$
    **end if**
**end function**

function identifies all directly connected nodes to a given node, essentially determining potential next hops in the packet's journey towards the gateway. Node availability is crucial for reliable packet forwarding, assessed by the `NodeIsAvailable` function, which evaluates a node's operational status based on a probability of being already processing another packet. This probability is computed through a Poisson process, as the Aloha case. Considering the data rate of the devices $\lambda$ as the Poisson process rate, we estimate the probability of $k$ arrivals at node $i$ in the vulnerable interval of $2T$, given $k = 0$. That probability is compared with a random generated threshold to simulate network dynamic uncertainty. In subsection 4.1 we compare using just default route, waiting when a next node is not available. A jupyter notebook with the project implementations is available on Github.

# 4 Experimental Results

## Experimental Setup

Key simulation parameters include:

- **Metrics:** Round Trip Time (RTT), Waiting Time, Success Rate, Cumulative Deadlocks.
- **Algorithms:** $\epsilon$-greedy, Upper Confidence Bound (UCB).
- **Network Topologies:** Uniform Grid (fully and partially connected), Random Network.

Simulation parameters are given in Appendix A.

### Network Topologies

Three network topology instances are considered:

1. **Uniform Grid (Fully Connected):** A 5x5 grid where each node is fully connected to all others, serving as a baseline for maximal connectivity.
2. **Uniform Grid (Partially Connected):** A 5x5 grid with reduced connectivity to simulate more realistic scenarios.
3. **Random Network:** Nodes are randomly connected, ensuring at least one path to the gateway per node.

See Appendix A, for visual examples of topology.

## 4.1 $\epsilon$-Greedy vs Default Routing

Figure 1 illustrates the significant impact of the hybrid $\epsilon$-greedy next-hop policy on reducing the waiting time across all three network topologies, resulting in the lowest average RTT. This approach ensures deadlocks occur with very low probability, as explained in subsection 3.2.3, attributing to the hybrid nature of the algorithm. In contrast, default routing, which relies on predetermined next-hop decisions, can lead to increased waiting times due to potential node unavailability.

### 4.1.1 Exploration versus Exploitation

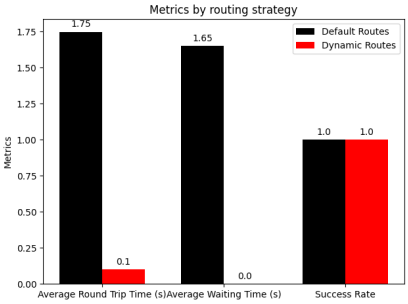
Figure 3 indicates that the optimal balance between exploration and exploitation is attained at $\epsilon = 0.7$. This finding suggests that, particularly in scenarios involving random graph topologies, a preference for exploration is beneficial as it leads to a reduction in the RTT metrics.

Further, an illustration of the numerical balance between exploration and exploitation is provided in Figure 2, within the context of a randomly structured network topology, at an $\epsilon$ value of 0.4.

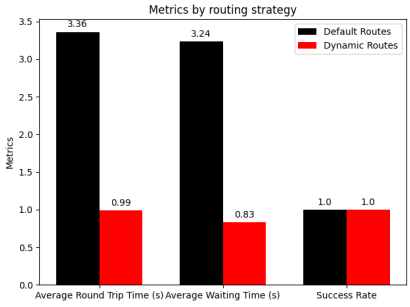
## 4.2 UCB-Based Heuristics

The UCB heuristic demonstrates robust performance, notably with the constraint of limited local knowledge, as illustrated in Figure 4. The average RTT is reported to be in the order of a few seconds, applicable to both partially and randomly connected graphs. The success rate reaches a peak of 92% in partially connected networks and 87% in randomly connected ones. Consequently, it is pertinent to examine the rate of deadlock occurrences. Interestingly, the deadlock growth rate decreases as the number of simulations increases, allowing the cumulative metric to converge, as shown in Figures 4d and 4b.

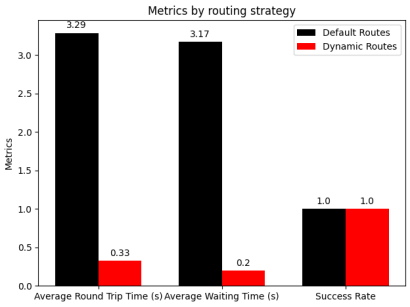
Despite a higher average RTT compared to the $\epsilon$-greedy strategy, the UCB heuristic exhibits comparable computational times when operating on both partially and randomly connected graphs, as detailed in table 2. It is important to note in interpreting these results that the $\epsilon$-greedy strategy benefits from precomputed shortest paths, whereas the UCB algorithm constructs paths dynamically.




(a) Metrics on a fully connected graph




(b) Metrics on a partial connected graph




(c) Metrics on a random connected graph

Figure 1: $\epsilon$-greedy strategy - Comparative metrics on different graph topologies

## 4.3 Algorithm Performance Comparison

The performance of different routing algorithms across various network topologies is summarized in the table below 2. Cumulative computational time is computed given the total number of simulations.

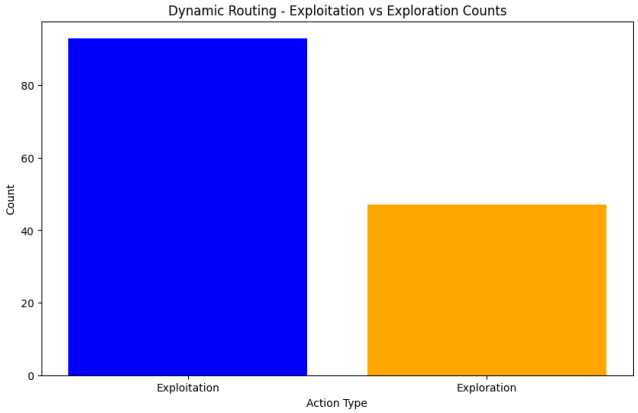



Figure 2: Exploitation vs Exploration a random connected graph.

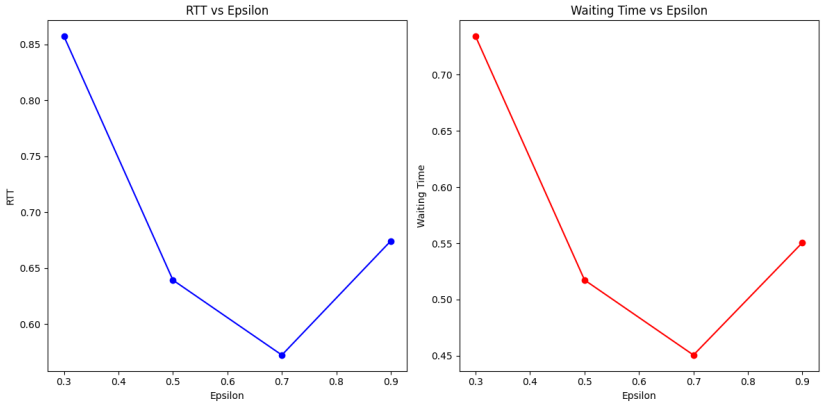


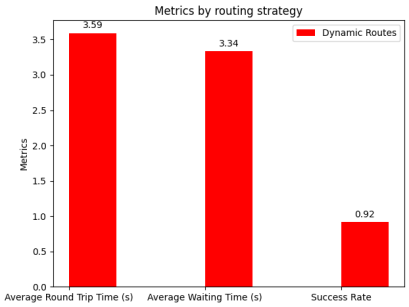

Figure 3: $\epsilon$ tuning vs RTT on a random connected graph. $\epsilon \in \{0.3, 0.5, 0.7, 0.9\}$.

| Algorithm | Network Topology | Number of Experiments | Total Computational Time |
|---|---|---|---|
| $\epsilon$-Greedy | Fully Connected Grid | 300 | 14ms |
| $\epsilon$-Greedy | Partially Connected Grid | 300 | 4m 10s |
| $\epsilon$-Greedy | Random Network | 300 | 1m |
| UCB | Partially Connected Grid | 300 | 16m 41s |
| UCB | Random Network | 300 | 7m 16s |

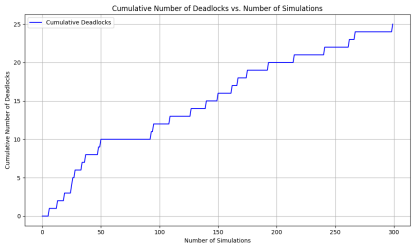
Table 2: Algorithm Computational Time Comparison.

# 5 Conclusion

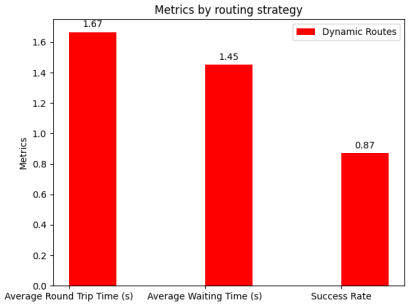
In this study, we explored the optimization of routing algorithms within WSNs characterized by various network topologies, including fully connected, partially



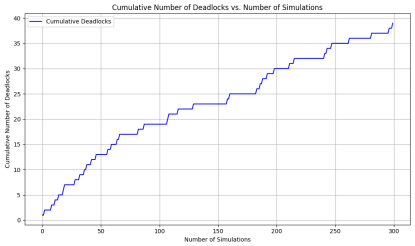

(a) UCB Metrics on a partial connected graph




(b) Deadlocks cumulative rate on a partial connected graph




(c) UCB Metrics on a random connected graph




(d) Deadlocks cumulative rate on a random connected graph

Figure 4: Comparative metrics and deadlock rates in partial and random connected graphs using UCB.

connected grids, and random networks. Through extensive simulations, we examined the performance of the $\epsilon$-greedy and UCB based heuristic algorithms for routing adaptation, focusing on metrics such as RTT, waiting times, success rates, and occurrences of deadlocks.

Overall, the $\epsilon$-greedy algorithm demonstrated superior performance, leveraging extensive neighborhood knowledge that an end-node could exploit. However, this advantage comes at the cost of significant traffic overhead during the initial booting phase or when adding a new end-node to the network topology. This scenario renders the $\epsilon$-greedy policy particularly suitable for networks characterized by high duty cycling and availability, thus requiring hardware capabilities to maintain a local memory for routing decisions.

Conversely, the UCB algorithm showed robust performance, particularly because paths are built on-the-go without a pre-initiation phase to establish default paths. Upon initialization, nodes are merely required to discover their neighborhood to list all the reachable next hops. The UCB-based routing heuristic becomes especially appealing for low-memory IoT networks that could benefit from a moderate learning phase before actual deployment. This allows for path

rewarding and penalization, thereby enforcing more efficient routing paths.

Future extensions of this study could investigate fully decentralised and autonomous learning sensor networks within which multiple simultaneous transmissions can occur during a simulation. Evaluating the robustness of heuristics in handling multiple nodes attempting to perform an uplink transmission to the gateway, in a synchronised or pseudo-random fashion, would provide insights closer to real-world applications. For example, in the context of LoRaWAN, the first scenario might encompass Class B or C devices, while the second is closer to Class A devices.

A further line of evolution could examine the effects on UCB convergence when faced with unconstrained rewards, which might span a continuous range in $\mathbb{R}$ or assume deterministic values in $\mathbb{Z}$, extending beyond the conventional range of $[0, 1]$. From the perspective of heuristics, future extensions could explore adversarial mechanisms such as EXP3 [21], or delve into more sophisticated policy gradient algorithms. These algorithms could prove particularly suitable for large-scale deployment in environments characterized by a complex state space. This complexity is underscored by challenges such as multiple concurrent transmissions, broadcast extensions, and multiple gateways. Such conditions necessitate a more intricate agent capable of navigating time-sensitive networks efficiently.

# References


[1] M. S. Aslam, A. Khan, A. Atif, S. A. Hassan, A. Mahmood, H. K. Qureshi, and M. Gidlund, "Exploring multi-hop lora for green smart cities," *IEEE Network*, vol. 34, no. 2, pp. 225–231, 2019.

[2] C.-H. Liao, G. Zhu, D. Kuwabara, M. Suzuki, and H. Morikawa, "Multi-hop lora networks enabled by concurrent transmission," *IEEE Access*, vol. 5, pp. 21 430–21 446, 2017.

[3] L. Prade, J. Moraes, E. de Albuquerque, D. Rosário, and C. B. Both, "Multi-radio and multi-hop lora communication architecture for large scale iot deployment," *Computers and Electrical Engineering*, vol. 102, p. 108242, 2022.

[4] S. Barrachina-Muñoz, B. Bellalta, T. Adame, and A. Bel, "Multi-hop communication in the uplink for lpwans," *Computer Networks*, vol. 123, pp. 153–168, 2017.

[5] M. Anedda, C. Desogus, M. Murroni, D. D. Giusto, and G.-M. Muntean, "An energy-efficient solution for multi-hop communications in low power wide area networks," in *2018 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*. IEEE, 2018, pp. 1–5.

[6] M. S. Iqbal, G. W. Wiriasto *et al.*, "Multi-hop uplink for low power wide area networks using lora technology," in *2019 6th International Conference on Information Technology, Computer and Electrical Engineering (ICITACEE)*. IEEE, 2019, pp. 1–5.

[7] W. Zhou, Z. Tong, Z. Y. Dong, and Y. Wang, "Lora-hybrid: A lorawan based multihop solution for regional microgrid," in *2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS)*. IEEE, 2019, pp. 650–654.

[8] J. Dai, X. Li, S. Han, Z. Liu, H. Zhao, and L. Yan, "Multi-hop relay selection for underwater acoustic sensor networks: A dynamic combinatorial multi-armed bandit learning approach," *Computer Networks*, p. 110242, 2024.

[9] A. A. Al Islam, S. I. Alam, V. Raghunathan, and S. Bagchi, "Multi-armed bandit congestion control in multi-hop infrastructure wireless mesh networks," in *2012 IEEE 20th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*. IEEE, 2012, pp. 31–40.

[10] J. Boyan and M. Littman, "Packet routing in dynamically changing networks: A reinforcement learning approach," *Advances in neural information processing systems*, vol. 6, 1993.

[11] J. Zhang, J. Tang, and F. Wang, "Cooperative relay selection for load balancing with mobility in hierarchical wsns: A multi-armed bandit approach," *IEEE Access*, vol. 8, pp. 18 110–18 122, 2020.

[12] J. Zhu, Y. Song, D. Jiang, and H. Song, "Multi-armed bandit channel access scheme with cognitive radio technology in wireless sensor networks for the internet of things," *IEEE access*, vol. 4, pp. 4609–4617, 2016.

[13] P. Zhang, A. Y. Gao, and O. Theel, "Bandit learning with concurrent transmissions for energy-efficient flooding in sensor networks," *EAI Endorsed Transactions on Industrial Networks and Intelligent Systems*, vol. 4, no. 13, 2018.

[14] S. Henri, C. Vlachou, and P. Thiran, "Multi-armed bandit in action: Optimizing performance in dynamic hybrid networks," *IEEE/ACM Transactions on Networking*, vol. 26, no. 4, pp. 1879–1892, 2018.

[15] G. Oddi, A. Pietrabissa, and F. Liberati, "Energy balancing in multi-hop wireless sensor networks: an approach based on reinforcement learning," in *2014 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. IEEE, 2014, pp. 262–269.

[16] X. Liang, I. Balasingham, and S.-S. Byun, "A multi-agent reinforcement learning based routing protocol for wireless sensor networks," in *2008 IEEE International Symposium on Wireless Communication Systems*. IEEE, 2008, pp. 552–557.

[17] M. Coupechoux, "Random access lecture slides," https://moodle.polytechnique.fr/pluginfile.php/601843/mod_resource/content/1/03-randomaccess.pdf.

[18] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, pp. 235–256, 2002.

[19] S. Bubeck, "Jeux de bandits et fondations du clustering," http://sbubeck.com/Bubeckthesis.pdf.

[20] O. Cappé, A. Garivier, O.-A. Maillard, R. Munos, and G. Stoltz, "Kullback-leibler upper confidence bounds for optimal sequential allocation," *The Annals of Statistics*, pp. 1516–1541, 2013.

[21] https://www.computer.org/csdl/proceedings-article/focs/1995/71830322/12OmNAYGlx6.

[22] "Iot protocols lecture, slide 37," https://moodle.polytechnique.fr/pluginfile.php/601851/mod_resource/content/2/09-iotprotocols.pdf.

# A Appendix

## Simulation Parameters

The simulation is initialized with the following parameters:

| Parameter | Value |
|---|---|
| Number of Nodes | 30 |
| Communication Radius[1](Fully Connected) | 100 |
| Communication Radius (Partial and Random Connected) | 70 |
| Area Size | (300, 300) |
| Gateway Node | 30 |
| Number of Simulations | 300 |
| Timeout | 5s |
| $\epsilon$ | 0.4 |
| $\lambda$ Rate | 5.5 [22] |
| Transmission Time | 36.6 ms [22] |

Table 3: Simulation Parameters

[1]In the simulation scenario, Communication Radius is an arbitrary unit used to play with node connectivity degree.

## Example of Network Topologies

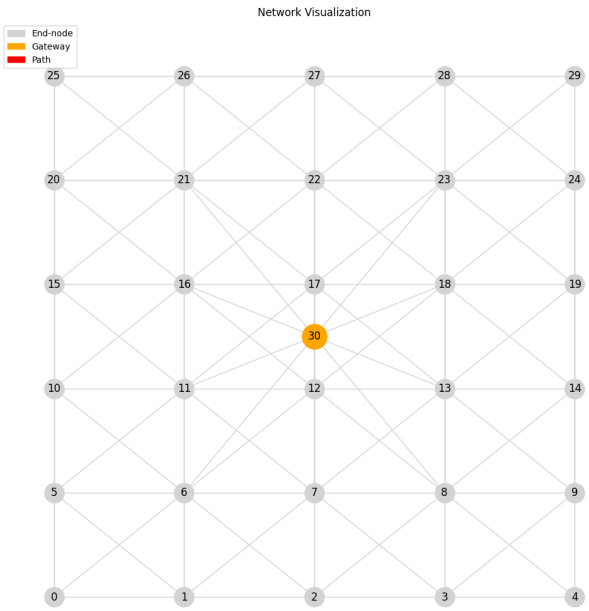



Figure 5: Fully connected network topology.

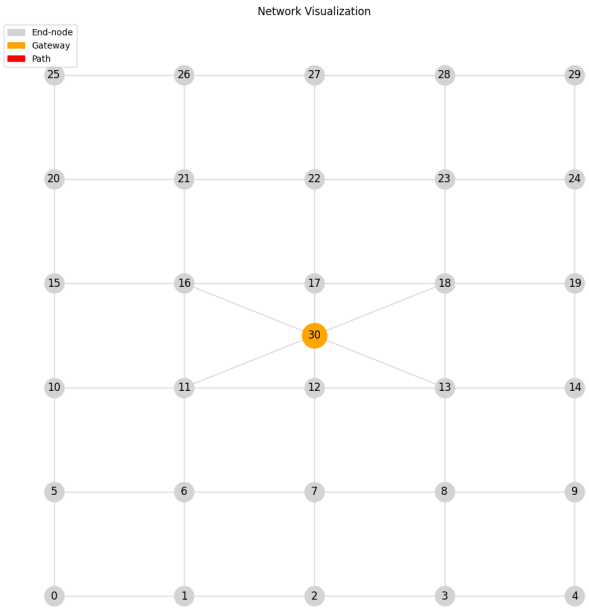



Figure 6: Partially connected network topology.

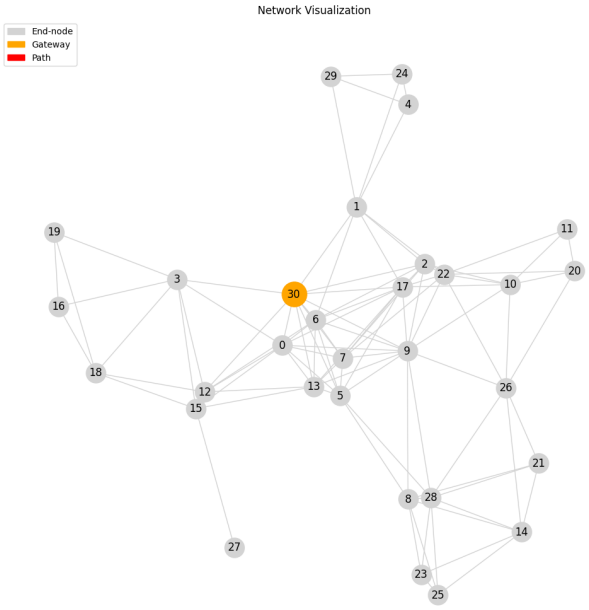



Figure 7: Randomly connected network topology.