# Multicommodity Flow Reliability for Energy Harvesting Wireless Sensor Networks

John Penaflor and Mohammed Elmorsy *

Department of Computer Science, MacEwan University, Edmonton, Canada;
Email: penaflorj2@mymacewan.ca (J.P.)
*Correspondence: elmorsym@macewan.ca (M.E.)

***Abstract*—This paper considers energy harvesting wireless sensor networks (EH-WSN) with multiple sinks supporting concurrent applications. Each application is associated with a set of sensor nodes that generate and send traffic to the associated application sink. Each node can relay any application traffic toward the application sink. In addition, each node uses an energy management unit to control the amount of traffic the node can relay based on its available energy. With the nodes' energy levels fluctuations, it is essential to quantify the network's ability to fulfill the different applications' quality of information and service requirements. Therefore, a novel multicommodity flow reliability problem (called *MultiFlowRel*) is formalized to estimate the likelihood that at least a certain amount of each application traffic is delivered to the associated application sink. The proposed problem is proven to be #P-hard, and an iterative bounding framework is proposed for deriving lower bounds on the exact reliability solutions. The proposed framework compute's exact reliability solutions if allowed a sufficient number of iterations. Numerical results show the effectiveness of using the proposed solution to obtain good lower reliability bounds and exact solutions in reasonable running times. Furthermore, the results show examples of the use of the proposed framework in solving some interesting network design problems (e.g. optimal sink locations and appropriate transmission parameters).**

***Keywords*—energy harvesting wireless sensor network, network reliability, energy management, multipurpose wireless sensor networks, iterative methods, probabilistic graphical models**

## I. Introduction

The use of wireless sensor networks has received considerable preference in many applications, e.g., health monitoring, environmental and industrial applications. A wireless sensor network (WSN) consists of sensor nodes with limited energy resources that sense some of interest phenomena from the surrounding environment and send the sensed data to one or more sink nodes in the network. Extensive research work on WSNs has been done over the past decade. This research is diversified and includes many directions, such as: using localization techniques for estimating sensor nodes' locations (e.g. [1–4]), using network deployment techniques for enhancing the wireless sensor network quality of information and services (e.g., [5, 6]), using energy harvesting techniques (e.g., [7–10]) and energy provisioning techniques (e.g., [11, 12]) for constructing WSNs with prolonged lifetime. In an energy harvesting WSN (denoted EH-WSN), each node harvest energy from the surrounding ambient sources (e.g., harvesting solar and wind energies) and use the harvested energy to recharge its battery. In addition, each node can use energy provisioning techniques to optimize its energy consumption and balance the energy use among the network nodes. Energy provisioning techniques introduced in the literature include energy efficient routing protocols (e.g., [11], [13–22]), energy efficient scheduling protocols (e.g., [23]) and topology control techniques (e.g., [24, 25]).

As a brief discussion about some recent research work for EH-WSNs, Nguyen *et al.* [26] proposed an energy-aware routing protocol for EH-WSNs. In their work, routing decisions are based on each node's available energy and distance from the sink. In addition, each node adjusts the active/sleep scheduling of its communication module based on its available energy level. The authors of [27, 28], proposed energy-aware opportunistic routing protocols for EH-WSNs. In an opportunistic routing protocol, nodes are unaware of the network topology. Instead, each node utilizes the broadcast nature of the wireless medium that allows adjacent nodes to receive copies of the node's sent traffic. In opportunistic routing, when a node attempts to deliver a packet to the sink, it broadcasts its packet to its adjacent nodes. After that, one of these adjacent nodes is selected based on some computed priority factor to forward the received packet toward the sink. Shafieirad *et al.* [27] proposed that the priority factor for each adjacent node is calculated based on the node's available energy and the distance between this node and the sink. Li *et al.* [28] proposed a Long short-term memory (LSTM) solar prediction model that allows each node to predict its harvested energy. The priority factor of each adjacent node is estimated based on the current node's available energy, the predicted harvested

Manuscript received December 9, 2022; revised May 26, 2023, accepted June 20, 2023.

energy, the node distance from the sink and the packet delivery ratio between the originating node and this node.

Another recent research direction considers WSNs shared between multiple applications. In such a network, WSN nodes serve multiple applications' requests while aiming to meet each application-defined quality of service and information metrics. For such networks, some recent researches aim to construct resource allocation and network management frameworks (e.g., [29]). As a brief review of some of the introduced work in this direction, Yıldırım and Tatar [30] considered using middleware server systems (MBSS) to process incoming application requests. The authors proposed a software framework called Firat Virtual WSN framework (FVWSN) that runs in MBSSs, collects the uploaded client application logic and allocates sensor nodes' resources for serving each application request. Delgado *et al.* [31] proposed an application admission and resource allocation framework. The proposed framework allocates network resources for application requests while achieving the defined applications' sensing coverage requirements. In addition, the proposed framework considers sensor node level constraints (e.g., processing, energy and storage capabilities) and network constraints (e.g., bandwidth of communication links and routing protocols). Hajian *et al.* [32] proposed a software-defined networking-based routing protocol for achieving load balancing across sensor nodes that serves multiple applications. Gao *et al*. [33] defined a data-sharing problem between various applications. Each application requests data sampling intervals from sensor nodes. The requested sample intervals can have some overlapping. The problem calls for finding common sampling intervals between the application requests. Therefore, the problem aims to minimize the generated and sent traffic across the network. The authors propose a greedy framework for solving their defined problem.

This paper considers energy harvesting wireless sensor networks shared between multiple applications. To quantify the ability of such networks in fulfilling the different applications' quality of service and information requirements, a novel flow reliability problem (called *MultiFlowRel)* is formalized. In this problem, each application has a set of sensor nodes and an associated sink node. Therefore, the considered network contains multiple sinks. Each application sensor node periodically generates traffic and sends it to the corresponding application sink. Each node runs independently an energy management unit that allows the node to adjust its transmission capacity based on its available energy. The node transmission capacity represents the maximum amount of traffic the node can send in a certain time interval. The problem calls for estimating the likelihood that each application sink node receives a certain amount of its application sensor nodes' generated traffic. To the best of the authors' knowledge, the proposed work is the first to discuss a flow reliability problem for multi-sink energy harvesting wireless sensor networks shared between multiple applications. As a summary of some related literature work, Oteafy and Hassanein [34] proposed a resource-sharing framework for wireless sensor networks shared between various applications. Chakraborty *et al.* [35] consider a network reliability problem for wireless sensor networks with a single sink. Each node is assumed to have unlimited transmission capacity and can be either sensing and communicating, communicating only or failing. The considered problem calls for estimating the likelihood that a certain amount of traffic arrives at the sink. The authors propose an exact algorithm for solving the considered problem. Elmorsy and Elmallah [36] formalized a flow reliability problem for energy harvesting wireless sensor networks with a single sink node and a single application utilizing the network resources. The authors propose an iterative framework for deriving lower bounds to the exact reliability solutions.

Below is a summary of the proposed contributions in this paper:

(1) A novel flow reliability problem (called *MultiFlowRel*) is formalized and proven to be #P-hard.
(2) An iterative framework that utilizes special structures, known as pathsets, is proposed for deriving lower bounds to the exact reliability solutions.
(3) A key ingredient in the proposed iterative framework is obtaining a set of high probable pathsets. Therefore, the optimal extension to pathset (denoted *E2P*) problem is formalized to construct high probable pathsets. The *E2P* problem is proven to be NP-hard.
(4) A heuristic algorithm is proposed to solve the proposed *E2P* problem.
(5) The obtained numerical results show the effectiveness of the proposed iterative framework and how it can solve interesting network design problems.

The rest of the paper is organized as follows. Section II formalizes the *MultiFlowRel* problem and introduces some solution concepts for constructing the proposed iterative framework. Section III presents an overview of the proposed iterative framework. Section IV formalizes the *E2P* problem and the proposed heuristic for solving this problem. Lastly, Section V presents the obtained numerical results.

## II. Problem Formulation

In this section, the *MultiFlowRel* problem is formalized. In addition, some of the concepts used in developing a solution are introduced.

### A. System Model

An energy harvesting wireless sensor network (EH-WSN) is considered that utilizes a time-slotted model where time is divided into equal time slots. In each time slot, each node uses a flow-based energy management scheme to maximize the node's lifetime. At the start of each time slot, the energy management unit of each node independently estimates its energy level and decides the maximum number of packets the node can transmit during this time slot. A node's chosen maximum allowed number of transmitted packets includes the node's generated packets and the relayed packets received from other nodes. A node *a* can communicate with node *b* if *b* lies within *a*'s transmission range.

To model a node's behaviour, each node *n* is assumed to exist in one of the possible energy states (denoted $ES = \{s_1, s_2, \cdots, s_m\}$) in each time slot based on its energy level. Each energy state $s_i$ for each node *n* has a corresponding maximum number of transmitted packets denoted $cap_{out}(n, s_i)$ that includes both node *n* generated and relayed packets. For example, a node *n* exists in state $s_1$ in a time slot if the node's energy level is within [80%, 100%] of its energy stored capacity with a corresponding $cap_{out}(n, s_1) = 10$ packets that *n* can, at most, transmit during this time slot. The probability that node *n* exists in an energy state $s_i$ is denoted $p(n, s_i)$. Therefore, $\sum_{i=1}^{m} p(n, s_i) = 1$ where $m$ is the number of possible energy states for node *n*. Such probabilities can be estimated by running experiments for the considered network over a WSN simulator where time is divided into suitable time slots. Then for each node *n* and each possible state $s_i$, $p(n, s_i)$ is estimated as the number of time slots, in which node *n* exists in state $s_i$, divided by the total simulation time.

The considered EH-WSN is assumed to be shared between a set of applications denoted $APP = \{app_1, app_2, \ldots, app_k\}$. Each application $app_i$ reserves a set of sensor nodes (denoted $APNodes(app_i)$) and a sink node (denoted $APsink(app_i)$). Each sensor node can be at most associated with a single application. However, some nodes may not be associated with any application. Each sensor node *n*, associated with application $app_i$, generates an amount of data packets denoted $gdata(n)$, and sends them to the application sink node $APsink(app_i)$. Each node in the network (including applications' associated nodes and sinks) can relay traffic from any source node toward its destination sink node in each time slot. To satisfy each application $app_i$ requirements, at least a specified number (denoted $K_{req}(app_i)$) of data packets should be delivered from $APNodes(app_i)$ nodes to $app_i$ sink node. The following formalizes the proposed *MultiFlowRel* problem.

**Definition (the *MultiFlowRel* problem):** Let an EH-WSN represented by a probabilistic directed graph $G = (V \cup Sinks, E)$ that uses a time slotted model and is shared between a set of applications $APP = \{app_1, \cdots, app_k\}$. In each time slot, each node $n \in (V \cup Sinks)$ exists in an energy state $s_i \in ES$ set of possible energy states with probability $p(n, s_i)$ where $n$ can transmit at most $cap_{out}(n, s_i)$ packets. Each application $app_i$ has a set of associated nodes denoted $APNodes(app_i) \subset V$ and an associated sink denoted $APsink(app_i) \in Sinks$. In each time slot, each node $n \in APNodes(app_i)$ generates $gdata(n)$ packets to be transmitted to $APsink(app_i)$. The problem calls for finding the probability denoted $MFRel(G, APNodes, APsink, P, gdata, cap_{out}, K_{req})$ that the network is in a state where for each application $app_i$, at least a defined amount $K_{req}(app_i)$ of data packets are delivered from $APNodes(app_i)$ to $APsink(app_i)$.

**Example 1:** Fig. 1 shows an instance of the *MultiFlowRel* problem where the network is shared between 2 applications (denoted $app_1$ and $app_2$) with the following assumptions:

- The first application has sensor nodes ($APNodes(app_1)$) 8 and 9. Also, its sink node ($APsink(app_1)$) is 1 and $K_{req}(app_1) = 10$. Nodes 8 and 9 generate 5 and 6 data packets in each time slot.
- The second application has sensor nodes ($APNodes(app_2)$) 3 and 6. Also, its sink node ($APsink(app_2)$) is 7 and $K_{req}(app_2) = 4$. Nodes 3 and 6 generate an equal amount of data packets $gdata(3) = gdata(6) = 4$.
- $ES = \{s_{high}, s_{low}, s_{fail}\}$
- Each node $n \in (V \cup Sinks)$ has the following states' information: $\{(p(n, s_{high}) = 0.25, cap_{out}(n, s_{high}) = 16), (p(n, s_{low}) = 0.5, cap_{out}(n, s_{low}) = 8), (p(n, s_{fail}) = 0.25, cap_{out}(n, s_{fail}) = 0)\}$.

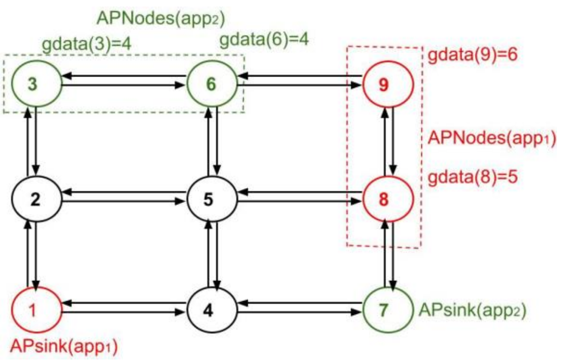


Figure 1. An instance of the *MultiFlowRel* problem.

**Theorem 1.** *MultiFlowRel* is #P-hard

**Proof.** AboElFotoh and Colbourn [37] shows that the two-terminal reliability problem (*2REL*) is #P-Complete even if it is restricted to partial grid networks with equal nodes' communication ranges and operating probabilities. An instance of the *2REL* problem consists of a probabilistic graph $G = (V, E)$ that contains two distinct vertices *s* and *t*. Each vertex $v \in V$ has an operating probability *p*. The *2REL* problem calls for computing the probability (denoted $Rel(G, s, t)$) that the network $G$ is in a state with at least one operating path from *s* to *t*. To prove that the *MultiFlowRel* problem is #P-hard, a polynomial time reduction from any instance $(G, s, t)$ of the *2REL* problem to an instance $(G, APNodes, APsink, P, gdata, cap_{out}, K_{req})$ of the *MultiFlowRel* problem is proposed. The reduction works as follows:

(1) Create the graph $G = (V, E)$ of the *MultiFlowRel* instance with all vertices and edges of the *2REL* problem instance graph.
(2) For each vertex $v \in V$ of the *MultiFlowRel* instance, create two possible energy states with the following capacity-probability pairs: $(1, p)$ and $(0, 1-p)$ where $p$ is the node operating probability in the *2REL* instance.
(3) For node $s$ in the constructed of *MultiFlowRel* problem, set $gdata(s)$ to 1.
(4) For each other node $n \neq s$ , set $gdata(n) = 0$.
(5) Create an application (denoted $app_1$ ) in the *MultiFlowRel* instance with the following information:
   - $APNodes(app_1) = \{s\}$
   - $APsink(app_1) = \{t\}$
   - $K_{req}(app_1) = 1$

The proof follows since $Rel(G, s, t) = MFRel(G, APNodes, APsink, P, gdata, cap_{out}, K_{req})$.

### B. Solution Concepts

Here, the solution concepts used in the proposed iterative framework are presented. These concepts include:

- **Network State.** In a network state, each node is assigned a state. Note that, for a network of size $N$ nodes where each node has $N_{ES}$ possible energy states, the number of possible network states is $(N_{ES})^N$ states. For example, if a network contains 6 nodes where each node has 3 possible states, the total number of possible network states is $3^6$ states.
- **Network Configuration.** In a network configuration (denoted as *C*), some nodes (but not necessary all) are assigned states. The probability that a configuration *C* arises is $Pr(C) = \prod_{(n,s_i)\in C} p(n, s_i)$ where $(n, s_i) \in C$.

Two network configurations are statistically disjoint (abbreviated as **s-disjoint**) **if** a node exists in both configurations with different states.

**Example 2.** If $C_1 = \{(1, s_{high}), (2, s_{low})\}$ and $C_2 = \{(1, s_{high}), (2, s_{high})\}$ , then $C_1$ and $C_2$ are **s-disjoint** configurations as node 2 exists in $C_1$ and $C_2$ with different states. Thus, $\Pr(C_1 \cup C_2) = \Pr(C_1) + \Pr(C_2)$.

- **Pathset.** a *MultiFlowRel* pathset is an operating configuration where for each application $app_i$, at least $K_{req}(app_i)$ data packets are delivered from $APNodes(app_i)$ nodes to $APsink(app_i)$ sink node.

**Example 3.** For the MultiFlowRel instance in Fig. 1, each node has three possible energy states $(s_{high}, s_{low}, s_{fail})$ with the following transmission capacities (16, 8, 0), respectively. The configuration $C = \{(9, s_{high}), (8, s_{low}), (6, s_{high}), (5, s_{high}), (4, s_{high}), (1, s_{high}), (7, s_{high})\}$ is a pathset since it guarantees that $K_{req}(app_1) = 10$ data packets are delivered from $APNodes(app_1)$ nodes to $APsink(app_1)$ sink, and $K_{req}(app_2) = 4$ data packets are delivered from $APNodes(app_2)$ nodes to $APsink(app_2)$ sink.

The main objective of the proposed iterative framework is to generate a set of **s-disjoint** pathsets and use it to derive lower bounds of the exact reliability solutions.

## III. Overview of the Iterative Framework

The proposed iterative framework generates a set of **s-disjoint** pathsets (denoted *P*) and uses them to derive lower bounds (denoted LBs) to the exact reliability solutions. Furthermore, the proposed framework compute's exact reliability solutions if sufficient iterations are allowed to generate a maximal set of **s-disjoint** pathsets. To generate a set of **s-disjoint** pathsets, the proposed framework uses the factoring method introduced in [38] to generate **s-disjoint** configurations from an input configuration. The framework starts by generating an empty configuration, extending it to a high probable pathset and marking the obtained pathset as unprocessed. Then, in each iteration, the framework selects the highest probable unprocessed pathset generated but not used in the previous iterations to generate other **s-disjoint** configurations. The framework then marks the selected pathset as processed and uses it to generate **s-disjoint** configurations. Lastly, the framework extends the generated configurations (if possible) to pathsets and marks the generated pathsets as unprocessed. The framework terminates after a defined number of iterations (denoted $N_{IT}$), or there is no unprocessed pathset to be processed. Lastly, the framework uses the generated set of **s-disjoint** pathsets (*P*) to drive lower bounds to the reliability solutions using the following equation:

$$\sum_{P_i \in P} \Pr(P_i) \leq MFRel$$

where $\Pr(P_i)$ is the probability of pathset $P_i$.

A key component in the proposed framework is how to extend an input configuration to a high probable pathset which is discussed in the next section.

## IV. Optimal Extension To Pathset

This section formalizes the optimal extension to pathset (denoted *E2P*) problem. In addition, a heuristic algorithm is presented to solve this problem.

### A. E2P Problem Definition

Given an instance $(G, APNodes, APsink, P, gdata, cap_{out}, K_{req})$ of the *MultiFlowRel* problem and a configuration *C*, the problem calls for finding a configuration $C_{new}$ such that: (1) $C \cup C_{new}$is a pathset, (2) and $C \cup C_{new}$ are node disjoint, and (3) $\Pr(C \cup C_{new})$ is as high as possible.

**Theorem 2.** *E2P* problem for *MultiFlowRel* problem is NP-hard.

**Proof.** To prove that *E2P* for *MultiFlowRel* problem is NP-hard, a reduction from the Exact Cover by 3 sets (denoted *X3C*) problem to the formulated *E2P* problem is proposed. The *X3C* problem [39] is an NP-complete problem. An instance of the *X3C* problem has two inputs: a set *X* of elements and a set of subsets *Y*. Each subset in *Y* contains exactly 3 elements of *X*. The size of the set *X* is *3q*, where *q* is a positive integer value. The *X3C* problem asks whether *Y* contains a collection *Y*' of *q* subsets from *Y* such that each element $x \in X$ occurs exactly in one subset of *Y*'.

The reduction from the $X3C(X,Y)$ instance to the proposed $E2P(G, APNodes, APsink, p, gdata, cap_{out}, K_{req}, c)$ problem instance works as follows:

(1) For every element x ∈ X, create a node x in the *E2P* instance with the following information:
- $gdata(x) = 1$
- two possible energy states with the following transmission capacities (1, 0) and probabilities $(prob, 1 - prob)$ respectively where *prob* is a real value and $0 < prob < 1$

(2) For every subset $y \in Y$, create a node $y$ in the *E2P* instance with the following information:
- $gdata(y) = 0$
- two possible energy states with the following transmission capacities (3, 0) and probabilities $(prob, 1 - prob)$, respectively

(3) Create an edge *(x, y)* in the *E2P* instance **if** $x \in y$ in the $X3C(X,Y)$ instance. Therefore, each node $y$ has exactly 3 children.

(4) Create a node (denoted β) with $gdata(\beta) = 0$ and one possible energy state with transmission capacity $= 3q$ packets and probability = 1 in the E2P instance.

(5) For each node y ∈ Y, create an edge *(y, β)*

(6) Create an application (denoted $app_1$) in the E2P instance with the following information:
- $APNodes(app_1)$ contains every node x that corresponds to an element in *X* in the *X3C(X, Y)* instance.
- $APsink(app_1) = \beta$
- $K_{req}(app_1) = 3q$

(7) Create an empty configuration C of the E2P instance.

Each node $y \in Y$ has exactly 3 children. In addition, all *X* nodes need to deliver their traffic to $\beta$. Therefore, the highest probable path set, that can be found if exists, has a probability $(prob)^{4q}$ since all X nodes should be assigned a state with transmission capacity = 1 and at least *q* nodes of *Y* nodes should be assigned a state with transmission capacity = 3. Therefore, the *X3C* problem instance has a solution **if** the reduced instance of the *E2P* problem has a solution with $\Pr(C \cup C_{new}) \geq (prob)^{4q}$.

The next subsection describes the proposed heuristic algorithm for solving the E2P problem.

### B. E2P Algorithm

Here, the proposed heuristic algorithm is presented for solving the *E2P* problem. As an overview, the proposed E2P algorithm works through the following phases:

P1) Transforming the *E2P* instance to an instance of the min cost multicommodity flow (denoted *MCMCF*) problem [40]

P2) Solving the constructed *MCMCF* problem instance

P3) Processing the *MCMCF* problem solution and constructing $C_{new}$

The proposed *E2P* algorithm considers the *MCMCF* problem with integer flows. The multicommodity flow problem with integer flows is shown in [39] to be NP-complete problem. An instance of the *MCMCF* problem [40] consists of a directed flow graph $G_{flow}(V,E)$ and a set of commodities *K*. Each edge $(a,b) \in E$ has an integer capacity $u(a,b)$ and a cost $c(a,b)$. Each commodity $K_i \in K$ is identified by the tuple $(g_i, t_i, d_i)$ where $g_i$ and $d_i$ are the source and destination respectively of commodity $K_i$, and $d_i$ is the integer demand that needs to be transmitted from $g_i$ to $t_i$. The *MCMCF* problem calls for determining for each edge $(a,b) \in E$ and each commodity $K_i \in K$, the amount of flow (denoted $f_i(a,b)$) from commodity $K_i$ that needs to be transmitted through *(a, b)* edge such that:

$$Minimize \sum_{(a,b)\in E}\left(c(a,b) \times \sum_{K_i\in K} f_i(a,b)\right) \quad (1)$$

$$\sum_{K_i\in K} f_i(a,b) \leq u(a,b) \quad \text{for } \forall(a,b) \in E \quad (2)$$

For $\forall a \in V$, $\forall K_i \in K$

$$\sum_{b\in V} f_i(a,b) - \sum_{b\in V} f_i(b,a) \begin{cases} d_i & a = g_i \\ -d_i & a = t_i \\ 0 & Otherwise \end{cases} \quad (3)$$

$$\forall \mathrm{f_i(a,b)} \in Z^{+} \cup \{0\} \quad (4)$$

Eq. (1) represents the objective function of the *MCMCF* problem, which is minimizing the total cost of delivering all commodities to their corresponding destinations. Eq. (2) states the capacity constraint where the total transmitted flow through an edge should not exceed the edge capacity. Eq. (3) states the flow conservation constraint. For each commodity, the difference between the amount of commodity flow leaving a node and the amount of commodity flow entering this node should be 0 except for this commodity's source and destination nodes. Eq. (4) enforces that the obtained solutions are non-negative integers.

Below, the details for each phase of the proposed *E2P* algorithm are discussed:

**(P1) Transforming** $E2P(G, APNodes, APsink, p, gdata, cap_{out}, K_{req}, C)$ **instance to** MCMCF $(G_{flow}, K)$ **instance**

(1) For each node/state pair $(a, s_i) \in C$, create in $G_{flow}$ two nodes (denoted $a_{in}$ and $a_{out}$). If $cap_{out}(a, s_i) > 0$, add an edge $(a_{in}, a_{out})$ with cost = 0 and capacity = $cap_{out}(n, s_i)$.

(2) For each node $b \in G$ and $b \notin C$ (free node):
- Create a set $ES(b)$ that contains all possible energy states of $b$ ordered ascendingly based on their transmission capacities. Therefore, if $s_i$ and $s_{i+1} \in$ ES, then $cap_{out}(b, s_i) < cap_{out}(b, s_{i+1})$.

- Create two nodes $b_{in}$ and $b_{out}$ in $G_{flow}$.
- For each energy state $s_i \in ES(b)$ with a transmission capacity $cap_{out}(b, s_i) > 0$, create an edge $(b_{in}, b_{out})$ with the following parameters:

$$\text{cost} = \lceil -1000 \ \log(p(b, s_i)) \rceil \tag{5}$$

$$capacity = \begin{cases} cap_{out}(b, s_i) & i = 1 \\ cap_{out}(b, s_i) - cap\_out(b, s_{i-1}) & i > 1 \end{cases} \tag{6}$$

The cost function Eq. (5) assigns lower costs for node/state pairs with higher probabilities. Therefore, node/state pairs with higher probabilities are given more preference while extending the input configuration $C$ to a pathset. For each node $b \notin C$, the capacity function Eq. (6) ensures that the total capacity of all edges between $b_{in}$ and $b_{out}$ is equal to the maximum transmission capacity of node $b$ in its energy states.

(1) For each edge $(a, b) \in G$, create an edge $(a_{out}, b_{in})$ in $G_{flow}$ with cost = 0 and capacity = $\infty$.

(2) For each application $app_i$ in the *E2P* instance:
- Create a node $\beta_i$ in $G_{flow}$.
- For each node $n \in APNodes(app_i)$, create an edge $(\beta_i, n_{in})$ in $G_{flow}$ with cost = 0 and capacity = $gdata(n)$.
- Create a commodity $K_i$ in the *MCMCF* instance with source = $\beta_i$, destination = $APsink(app_i)$ in and demand = $K_{req}(app_i)$

**(P2) Solving the constructed *MCMCF* problem instance**

The constructed *MCMCF* instance can be solved using one of the integer programming methods [41]. The *E2P* algorithm uses the Gurobi optimizer [42] to solve the constructed instance. The Gurobi optimizer uses the branch and cut method [41] for solving the formulated *MCMCF* problem.

**(P3) Processing the *MCMCF* problem solution and constructing $C_{new}$**

After solving the constructed *MCMCF* instance, the *E2P* algorithm does the following:

(1) If the constructed *MCMCF* problem has no solution, the *E2P* algorithm terminates as the input configuration $C$ can't be extended to a pathset.

(2) Otherwise, the *E2P* algorithm creates an empty configuration $C_{new}$. Then, for each node $b \notin C$ with non-zero total flow from $b_{in}$ to $b_{out}$ in $G_{flow}$:
- Select the highest probable state $s_{max}$ from the possible energy states of node $b$ such that $cap_{out}(b, s_{max}) \geq \sum_{K_i \in K} f_i(b_{in}, b_{out})$ where $\sum_{K_i \in K} f_i(b_{in}, b_{out})$ is the total amount of flow transferred from $b_{in}$ to $b_{out}$ in $G_{flow}$
- Add the node/state pair $(b, s_{max})$ to $C_{new}$.

The constructed $C_{new}$ is the required extension to configuration $C$ such that $C \cup C_{new}$ is a pathset.

**Example 4.** In example 1, consider an input configuration $C = \{(1, s_{high}), (2, s_{fail}), (4, s_{high}), (5, s_{fail}), (7, s_{high})\}$:

Phase 1 of the *E2P* algorithm constructs the *MCMCF* instance with two commodities $K_1 = (\beta_1, 1_{in}, 10)$ and $K_2 = (\beta_2, 7_{in}, 4)$. $G_{flow}$ of the *MCMCF* instance is presented in Fig. 2, where the capacity/cost pair is presented on each edge. Phase 2 solves the constructed *MCMCF* instance. Phase 3 investigates the free nodes $\{3, 6, 8, 9\}$ with non-zero flow in the MCMCF instance:

- Node 6 has a total flow of 4 units transferred from $6_{in}$ and $6_{out}$. The highest probable state with enough capacity to transfer this flow is $s_{low}$ with capacity = 8. Therefore, the pair $(6, s_{low})$ is added to $C_{new}$.
- Node 9 has a total flow of 9 units. Therefore, the node/state pair $(9, s_{high})$ is added to $C_{new}$.
- Node 8 has a total flow of 14 units. Therefore, the node/state pair $(8, s_{high})$ is added to $C_{new}$.

The obtained $C_{new} = \{(6, s_{low}), (8, s_{high}), (9, s_{high})\}$ which satisfies that $C \cup C_{new}$ is a pathset.

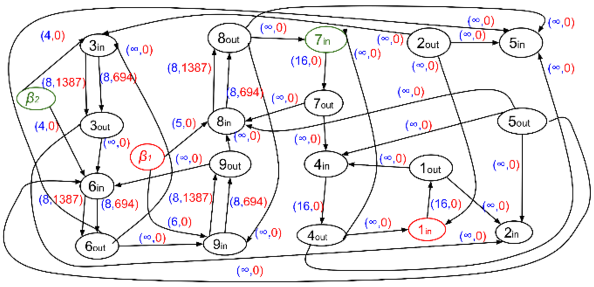


Figure 2. An example for $G_{flow}$ of a constructed MCMCF problem.

## V. Numerical Results

This section presents the numerical results that show the performance of the proposed framework and its use in solving interesting network design problems. The proposed framework is implemented in C++ using STL (Standard Template Library) classes and Gurobi Optimizer [42] for solving integer linear programming problems. An Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz server with 395 GBytes of memory is used. The experiments' maximum recorded memory size for running the *E2P* algorithm is about 10 megabytes. The length of the documented framework code is about 1950 lines.

The experiments are conducted on a $W \times L$ grid deployed networks. Each node's coordinates is denoted *(x, y)* where $x = 0, 1, \ldots, L-1$ and $y = 0, 1, \ldots, W-1$. Each node has $N_{ES}$ possible energy states with the following transmission capacities $\{0, \frac{MAXCAP}{N_{ES}-1}, \frac{2 \times MAXCAP}{N_{ES}-1}, \ldots, MAXCAP\}$ where $MAXCAP$ is varied during the experiments. Two applications (denoted $app_1$ and $app_2$) share the network resources. For $app_1$, the application sensor nodes ($APNodes(app_1)$) are the right-most column nodes except the bottom right corner node. The bottom left corner node is the sink node $APsink(app_1)$. For $app_2$, the application sensor nodes ($APNodes(app_2)$) are the top row nodes except the top right corner node. The sink node $APsink(app_2)$ is the bottom right corner node. Each application sensor node generates 12 data packets every time slot. The framework's default number of allowed iterations (denoted $N_{IT}$) is 5000.

### A. Exact Reliability Computations.

This set of experiments aims to investigate the effect of varying the network size and $N_{ES}$ on the running time of the proposed framework and the number of generated configurations (denoted $N_{configurations}$) while computing exact reliability solutions. The experiments are conducted on $W \times L$ grid deployed networks. Each node has a transmission range of 1.7 units and $N_{ES}$ possible energy states with $MAXCAP$ = 24 and equal probabilities. Therefore, if $N_{ES}$ = 3, the possible transmission capacities associated with the energy states are 0, 12 and 24 packets/slot. For each application $app_i$, $K_{req}(app_i)$ is set to half of the traffic generated by $app_i$ sensor nodes ($APNodes(app_i)$). Table I shows the obtained results for different network sizes where $N_{ES} =$ 3. Table II shows the obtained results while varying $N_{ES}$ for a 3×4 grid deployed network. The results in both tables show that:

- The number of generated network configurations ($N_{configurations}$) by the proposed framework while computing exact solutions is much less than the number of possible network states required by a brute force algorithm to obtain these solutions.
- Increasing the network size and $N_{ES}$ increases the running time of the proposed framework for obtaining exact solutions. The main reason is that more network configurations need to be processed, and the time required to run the *E2P* function for each generated configuration also increases.

TABLE I: EFFECT OF INCREASING NETWORK SIZE WITH $N_{ES}$ = 3

| **Network Size** | $N_{configurations}$ | **Number of Possible Network States** | **Running Time (Seconds)** |
|---|---|---|---|
| 2×2 | 31 | $3^4$ | 0.294325 |
| 2×3 | 87 | $3^6$ | 0.807948 |
| 3×3 | 1191 | $3^9$ | 12.027376 |
| 3×4 | 11631 | $3^{12}$ | 123.256963 |
| 4×4 | 209691 | $3^{16}$ | 2515.948289 |

TABLE II: EFFECT OF VARYING $N_{ES}$ NETWORK FOR 3×4 GRID DEPLOYED

| $N_{ES}$ | $N_{configurations}$ | **Number of Possible Network States** | **Running Time (Seconds)** |
|---|---|---|---|
| 2 | 73 | $2^{12}$ | 0.355533 |
| 3 | 11631 | $3^{12}$ | 123.256963 |
| 4 | 192595 | $4^{12}$ | 648.017521 |

### C. Effect of Varying Network Size on the Obtained Reliability Bounds

This set of experiments investigates the effect of varying the network size and the number of allowed iterations ($N_{IT}$) for the proposed framework on the obtained reliability lower bounds. The experiments are performed on $W \times W$ grid deployed networks where each node has a transmission range of 1.7 units. Each non-sink node has 3 possible energy states with associated transmission capacities (0, 12, 24) and probabilities (0.25, 0.25, 0.5), respectively. Sink nodes have one energy state with probability = 1 and a transmission capacity of 24 packets per slot. For each application $app_i$, $K_{req}(app_i)$ is set to a quarter of the total traffic generated by application nodes ($APNodes(app_i)$).

Results in Fig. 3 show that:

- Increasing the network size decreases the obtained reliability bounds as application sensor nodes become farther from their associated application sinks. Therefore, more nodes need to be included in the constructed pathsets with certain states, which decreases the probability of each obtained pathset.
- The gaps between the obtained bounds for the same network size decrease as $N_{IT}$ increases. In addition, the behaviour of the obtained curves for different $N_{IT}$ is similar. This shows the effectiveness of the proposed framework in obtaining good reliability bounds in a small number of iterations. In addition, this motivates the use of these bounds for solving interesting network design problems.

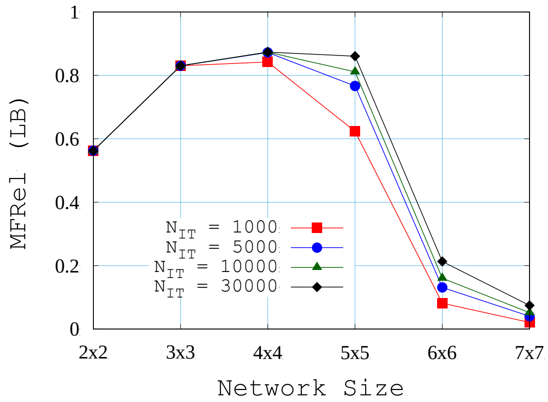


Figure 3. Effect of varying network size and $N_{IT}$.

### D. Effect of Varying Transmission Parameters on the Obtained Reliability Bounds

This set of experiments investigates the effect of varying each node's transmission range and capacity on the obtained reliability bounds. The experiments are conducted on $a$ 6 × 6 grid deployed network. Each node has a transmission range $r_{com}$, which varies during the experiments. Each non-sink node has 3 possible energy states with associated transmission capacities $(0, \frac{MAXCAP}{2}, MAXCAP)$ and probabilities (0.25, 0.25, 0.5), respectively. *MAXCAP* is varied during the experiments. Each sink node has one possible energy state with probability 1 and transmission capacity *MAXCAP* packets per slot. $K_{req}$ is set to 15 for each application.

The results in Fig. 4 show that increasing transmission range ($r_{com}$) and capacity increases the obtained reliability bounds. The main reason is that increasing the transmission range and capacity decreases the number of relay nodes required to deliver the traffic from application sensor nodes to their application sinks. Such results are important for network designers to choose node's transmission parameters that achieve at least certain reliability values.

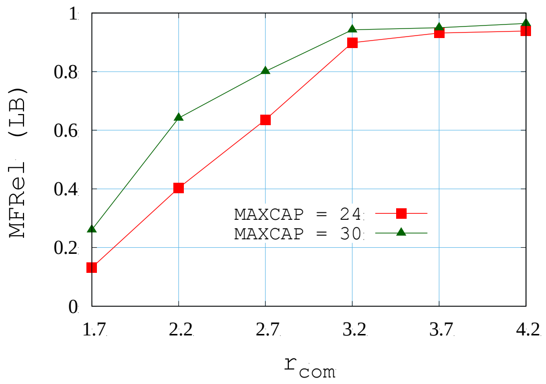


Figure 4. Effect of varying transmission parameters

### E. Optimal Deployment for Application Sinks

This experiment set shows how the proposed framework can be used to solve another network design problem. The considered problem calls for choosing the best sink location for each application among candidate locations to maximize reliability. The experiments are conducted on $6 \times 6$ grid deployed networks where each node has a transmission range of 1.7 units.

#### 1) Known traffic sources

In this set of experiments, each application sensor node generates 12 data packets every time slot. Each non-sink node has three possible energy states with associated transmission capacities (0, 12, 24) and probabilities $(\frac{1-p_{maxcap}}{2}, \frac{1-p_{maxcap}}{2}, p_{maxcap})$, respectively. $p_{maxcap}$ is varied during the experiments. Each sink node has one possible energy state with probability 1 and transmission capacity 24. Applications $app_1$ and $app_2$ have 5 candidate pairs of sinks' locations with coordinates $((i,i),(5-i,i))$ where $(i,i)$ and $(5-i,i)$ are the candidate locations for $app_1$ and $app_2$ sinks respectively; and $i \in [0,4]$. $K_{req}$ is set to 15 for each application.

Results in Fig. 5 show that the optimal sinks' locations for $app_1$ and $app_2$ are (4, 4) and (1, 4), respectively. Fig. 6 shows for each candidate pair of sink locations the number of application sensor nodes in each breadth-first search (BFS) layering from their corresponding application sink nodes. Placing $app_1$ and $app_2$ sinks in locations (4, 4) and (1, 4) allows application sensor nodes to have better BFS layering distribution from their corresponding sinks. Therefore, the number of required relay nodes for forwarding at least the required $K_{req}$ packets decreases, increasing the obtained reliability solutions.

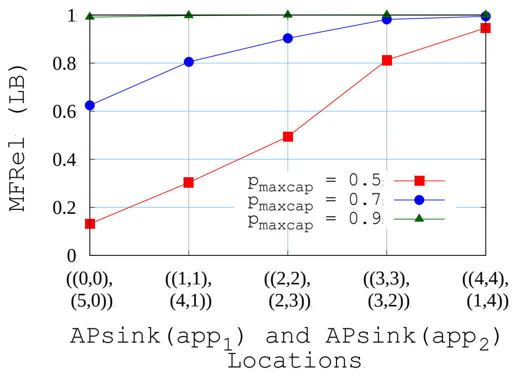


Figure 5. Effect of varying application sinks' locations with known traffic sources

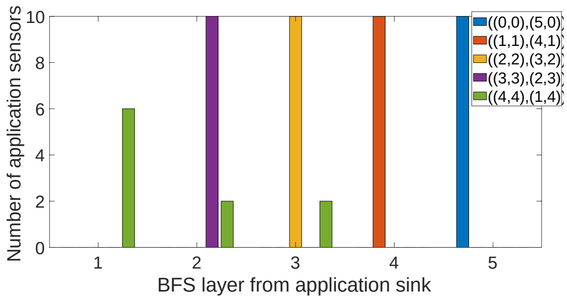


Figure 6. Application sensor nodes layering distribution from their sink nodes for known traffic source experiments

#### 2) Random traffic sources

In the second set of experiments, each application is associated with more sensor nodes. The sensor nodes associated with $app_1$include the top row and rightmost column nodes except corner nodes. The sensor nodes associated with $app_2$ include the bottom row and left-most column nodes except corner nodes. Only a known percentage of application sensor nodes (denoted $P_{source}$) generate traffic for each application, while the rest do not. The choice of application sensor nodes to generate traffic is random, where each chosen sensor node generates 12 data packets. For each application, $K_{req}$is set to the total amount of traffic generated by the application sensor nodes. Each non-sink node has three possible energy states with transmission capacities (0, 24, 48) and probabilities (0.25, 0.25, 0.5), respectively. Each sink node has one possible energy state with probability = 1 and transmission capacity = 48. Six candidate pairs of sink locations for $app_1$ and $app_2$ are investigated. Each candidate pair has $app_1$ and $app_2$ sink locations $((i,i),(5-i,i))$ where $i \in [0,5]$. Each experiment is repeated 20 times.

Fig. 7 shows the obtained results with 95% confidence. Both $((2,2),(2,3))$ and $((3,3),(3,2))$ pairs achieve the highest reliability compared to the other candidate pairs. Fig. 8 shows for each candidate pair of sink locations the number of application sensor nodes in each breadth-first search (BFS) layering from their corresponding application sink nodes. At location pairs $((2,2),(2,3))$ and $((3,3),(3,2))$, all sensor nodes are in BFS layers 2 and 3 from their corresponding sinks. At other candidate pairs, sensor nodes are distributed across more BFS layers from their corresponding sinks. Therefore, for location pairs ((2, 2), (2, 3)) and ((3, 3), (3, 2)), the number of relay nodes, required to guarantee the delivery of all the generated traffic from all application sensor nodes to their corresponding sink, is less which increases the obtained reliability.

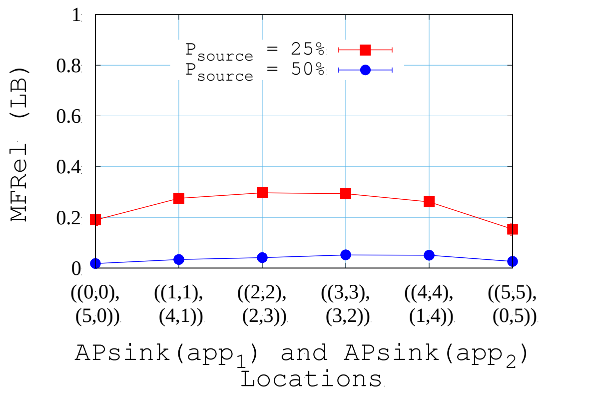


Figure 7. Effect of varying application sinks' locations with random traffic sources.

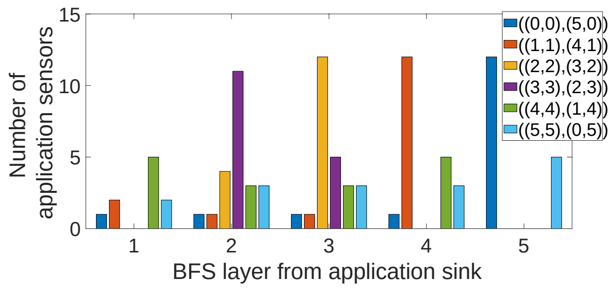


Figure 8. Application sensor nodes layering distribution from their sink nodes for random traffic source experiments.

## VI. Conclusion

This paper considers multi-sink energy harvesting wireless sensor networks that serve multiple concurrent applications. To adapt to the fluctuation of each node's available energy over time, each node controls its transmission capacity based on its available stored energy. A novel multicommodity flow reliability problem is formalized to estimate the likelihood that the shared network is in a state where all applications' requirements are met. The proposed problem is proven to be #P-hard. Therefore, a bounding framework is proposed for estimating lower bounds on the exact reliability solutions. The proposed framework produces exact solutions if allowed sufficient running time. Numerical results show the effectiveness of the proposed framework and its use in solving interesting network design problems.

Future research directions include developing effective algorithms for solving other network reliability problems for multi-purpose energy-harry wireless sensor networks and investigating the use of the developed algorithms in designing effective network management frameworks that achieve at least certain reliability levels.

## Conflict of Interest

The authors declare no conflict of interest.

## Author Contributions

John Penaflor conducted the research under the supervision of Dr. Mohammed Elmorsy. All authors contributed to the problem formulation, proofs, and proposed framework design. John Penaflor made the coding and testing that was validated by Dr. Mohammed Elmorsy. All authors contributed to the paper writing and checking. All authors had approved the final version.

## Funding

This research is supported by Riipen Level Up Project Funding, and MacEwan Undergraduate Student Research Initiative Project Funding.

## Acknowledgment

The authors wish to thank MacEwan University and Riipen for their support.

## References

[1] F. Mekelleche and H. Haffaf, "Classification and comparison of range-based localization techniques in wireless sensor networks," *Journal of Communications*, vol. 12, no. 4, pp. 221–227, 2017.

[2] O. Liouane, S. Femmam, T. Bakir, and A. B. Abdelali, "Improved two hidden layers extreme learning machines for node localization in range free wireless sensor networks," *Journal of Communications*, vol. 16, no. 12, pp. 528–534, 2021.

[3] J. Zhu, C. Lv, and Z. Tao, "An improved localization scheme based on dv-hop for large-scale wireless sensor networks," *Journal of Communications*, vol. 11, no. 12, pp. 1057–1065, 2016.

[4] V. Tam, K. Cheng, and K. Lui, "Using micro-genetic algorithms to improve localization in wireless sensor networks," *Journal of Communications*, vol. 1, no. 4, pp. 1–10, 2006.

[5] M. Farsi, M. A. Elhosseini, M. Badawy, H. A. Ali, and H. Z. Eldin, "Deployment techniques in wireless sensor networks, coverage and connectivity: A survey," *IEEE Access*, vol. 7, pp. 28940–28954, 2019.

[6] N. Rai and R. Daruwala, "A comprehensive approach for implementation of randomly deployed wireless sensor networks," *Journal of Communications*, vol. 14, no. 10, pp. 915–925, 2019.

[7] A. J. Williams, M. F. Torquato, I. M. Cameron, A. A. Fahmy, and J. Sienz, "Survey of energy harvesting technologies for wireless sensor networks," *IEEE Access*, vol. 9, pp. 77493–77510, 2021.

[8] D. Ma, G. Lan, M. Hassan, W. Hu, and S. Das, "Sensing, computing, and communications for energy harvesting IoTs: A survey," *IEEE Communications Surveys Tutorials*, vol. 22, no. 2, pp. 1222–1250, 2020.

[9] S. Sudevalayam and P. Kulkarni, "Energy harvesting sensor nodes: Survey and implications," *IEEE Communications Surveys Tutorials*, vol. 13, no. 3, pp. 443–461, 2011.

[10] A. A. Ka'bi, "Optimization of energy harvesting in mobile wireless sensor networks," *Journal of Communications*, vol. 17, no. 4, pp. 267–272, 2022.

[11] M. Warrier and A. Kumar, "Energy efficient routing in wireless sensor networks: A survey," in *Proc. 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, 2016, pp. 1987–1992.

[12] H. Yetgin, K. T. Cheung, M. El-Hajjar, and L. Hanzo, "A survey of network lifetime maximization techniques in wireless sensor networks," *IEEE Communications Surveys Tutorials*, vol. 19, no. 2, pp. 828–854, 2017.

[13] F. D. Tolba and P. Lorenz, "Multi path routing-based energy efficient for extending mobile wireless sensor networks lifetime," *Journal of Communications*, vol. 14, no. 12, pp. 1224–1228, 2019.

[14] F. Liao and W. Zhang, “An improved unequal clustering routing protocol for wireless sensor networks,” *Journal of Communications*, vol. 14, no. 2, pp. 148–152, 2019.

[15] N. K. Jain and A. Verma, “Relay node selection in wireless sensor network using fuzzy inference system,” *Journal of Communications*, vol. 14, no. 6, pp. 423–431, 2019.

[16] A. Rais, K. Bouragba, and M. Ouzzif, “Optimal routing and clustering technique for wireless sensor networks,” *Journal of Communications*, vol. 14, no. 9, pp. 758–764, 2019.

[17] D. A. Marenda, R. Muhammad, and N. R. Syambas, “Ring topology optimization for wireless sensor network: A new heuristic method,” *Journal of Communications*, vol. 13, no. 8, pp. 463–467, 2018.

[18] Y. Nie, S. Liu, Z. Chen, and X. Qi, “A dynamic routing algorithm for data-aggregation optimization in event-driven wireless sensor networks,” *Journal of Communications*, vol. 8, no. 8, pp. 521–528, 2013.

[19] R. Du, C. Ai, L. Guo, J. Chen, J. Liu, J. He, and Y. Li, “A novel clustering topology control for reliable multi-hop routing in wireless sensor networks,” *Journal of Communications*, vol. 5, no. 9, pp. 654–664, 2010.

[20] Y. Nie, S. Liu, Z. Chen, and X. Qi, “An adaptive state aware routing algorithm for data aggregation in wireless sensor networks,” *Journal of Communications*, vol. 8, no. 5, pp. 296–306, 2013.

[21] C. Zhang, “Cluster-based routing algorithms using spatial data correlation for wireless sensor networks,” *Journal of Communications*, vol. 5, no. 3, pp. 232–238, 2010.

[22] Y. Zhang and Q. Huang, “A learning-based adaptive routing tree for wireless sensor networks,” *Journal of Communications*, vol. 1, no. 2, pp. 12–21, 2006.

[23] L. Wang and Y. Xiao, “A survey of energy-efficient scheduling mechanisms in sensor networks,” *Mobile Networks and Applications*, vol. 11, no. 5, p. 723–740, 2006.

[24] P. Santi, “Topology control in wireless ad hoc and sensor networks,” *ACM Computing Surveys*, vol. 37, no. 2, p. 164–194, 2005.

[25] M. P. Lakshmi and D. P. Shetty, “Optimal algorithm for minimizing interference with two power levels in wireless sensor networks,” *Journal of Communications*, vol. 14, no. 12, pp. 1198–1204, 2019.

[26] T. D. Nguyen, J. Khan, and D. T. Ngo, “A distributed energy-harvesting-aware routing algorithm for heterogeneous IoT networks,” *IEEE Transactions on Green Communications and Networking*, vol. 2, no. 4, pp. 1115-1127, 2018.

[27] H. Shafieirad, R. S. Adve, and S. Shahbazpanahi, “Max-SNR opportunistic routing for large-scale energy harvesting sensor networks,” *IEEE Transactions on Green Communications and Networking*, vol. 2, no. 2, pp. 506–516, 2018.

[28] Y. Li, X. He, and C. Yin, “Energy aware opportunistic routing for energy harvesting wireless sensor networks,” in *Proc. 2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, 2020, pp. 1–6.

[29] I. Khan, F. Belqasmi, R. Glitho, N. Crespi, M. Morrow, and P. Polakos, “Wireless sensor network virtualization: A survey,” *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 553–576, 2016.

[30] G. Yıldırım and Y. Tatar, “Simplified agent-based resource sharing approach for WSN-WSN interaction in IoT/CPS projects,” *IEEE Access*, vol. 6, pp. 78077–78091, 2018.

[31] C. Delgado, M. Canales, J. OrtÅLın, J. GÅLallego, A. Redondi, S. Bousnina, and M. Cesana, “Joint application admission control and network slicing in virtual sensor networks,” *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 28–43, 2018.

[32] E. Hajian, M. R. Khayyambashi, and N. Movahhedinia, “A mechanism for load balancing routing and virtualization based on SDWSN for IoT applications,” *IEEE Access*, vol. 10, pp. 37457–37476, 2022.

[33] H. Gao, X. Fang, J. Li, and Y. Li, “Data collection in multi-application sharing wireless sensor networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 2, pp. 403–412, 2015.

[34] S. M. A. Oteafy and H. S. Hassanein, “Resource re-use in wireless sensor networks: Realizing a synergetic internet of things,” *Journal of Communications*, vol. 7, no. 7, pp. 484–493, 2012.

[35] S. Chakraborty, N. Goyal, S. Mahapatra, and S. Soh, “Minimal path-based reliability model for wireless sensor networks with multistate nodes,” *IEEE Transactions on Reliability*, vol. 69, no. 1, pp. 382–400, 2020.

[36] M. Elmorsy and E. S. Elmallah, “On flow reliability in energy harvesting wireless sensor networks,” in *Proc. IEEE International Conference on Communications (ICC)*, 2021, pp. 1–6.

[37] H. AboElFotoh and C. Colbourn, “Computing 2-terminal reliability for radio-broadcast networks,” *IEEE Transactions on Reliability*, vol. 38, no. 5, pp. 538–555, 1989.

[38] M. Elmorsy, E. Elmallah, and H. AboElFotoh, “On path exposure in probabilistic wireless sensor networks,” in *Proc. 38th Annual IEEE Conference on Local Computer Networks*, 2013, pp. 433–440.

[39] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*, W.H. Freeman, San Francisco, 1979.

[40] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, 1993.

[41] M. Conforti, G. CornuÅLejols, and G. Zambelli, *Integer Programming*, Springer International Publishing, 2014.

[42] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual.. [Online]. Available: https://www.gurobi.com