STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **x**, Month 2025, pp 0–19.
Published online in International Academic Press (www.IAPress.org)

Majority Voting Paraconsistent Annotated Logic Algorithm For Fault Tolerance And Detection In Wireless Sensor Networks

Abdullah Shawan alotaibi*

Department of Computer Science, College of Science and Humanities, Shaqra University, Saudi Arabia

Abstract Sensors in Wireless Sensor Networks (WSNs) are prone to faults. Since sensor nodes often share parts of their information, it becomes possible to detect inconsistencies between a neighbour's report and what is expected. In this work, such inconsistencies are analysed using the Paraconsistent Annotated Logic with Two Values (PAL2V) to compare the actual decisions of neighbouring nodes with the expected decisions reconstructed from the local, partial information each node possesses. An algorithm is proposed based on majority voting (MV-PAL2V), concluding a set of states describing the contradiction and guiding a specific corrective response to reduce the associated errors. Simulation results demonstrate that using PAL2V enhances the accuracy of majority voting, particularly by reducing false alarm rates and improving the detection of actual events. Moreover, the statistical and interaction analysis of the simulation results emphasised that the number of nodes plays a crucial role in the reliability of the WSNs' data.

Keywords Wireless Sensor Networks, Paraconsistent Annotated Logic, Fault Detection, Majority Voting, Machine Learning

DOI: 10.19139/soic-2310-5070-2837

1. Introduction

To meet the diverse needs of wireless sensor networks (WSNs) for various application fields, sensors are designed to be small, affordable, and energy-efficient. While these features make them suitable for a wide range of environments, a set of trade-offs has been faced, primarily limited communication range, reduced processing capabilities, and reduced sensor quality [1], [2], [3], [4]. Sensors can be randomly deployed in a special space for simplicity or deterministically to ensure coverage and connectivity [5]. The common activity patterns for sensors could be classified as: 1) Time-Driven, 2) Event-Driven, 3) Query-Driven, and 4) Hybrid-Driven; each of them might be more suitable for a specific situation and suffers a shortcoming for another. In the time-driven pattern, the sensor is asked to report the monitored property of interest according to an a priori defined schedule (i.e., property sensing time). However, it might face the challenge of high energy consumption and the discrete reporting of the measured property (i.e., it is not a continuous real-time reporting). The event-driven sensor reports only when a particular event has been triggered (i.e., a state-based or event-based sensor), so it might face the data redundancy challenge.

The event-driven pattern enables detection and reporting of events at the moment of occurrence, which can improve responsiveness in time-sensitive applications, as it can detect and report events when they happen [6], [7], [8]. The query-driven sensor is activated and transmits data only upon request or a specific query from the main controller or the central node. As such, it is beneficial for applications that require specific information to perform a task-oriented, sophisticated analysis, like patient monitoring. Moreover, query-driven is not based on predefined

*Correspondence to: Abdullah Alotaibi (Email: a.shawan@su.edu.sa). Department of Computer Science, College of Science and Humanities, Shaqra University, Saudi Arabia.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

states, events, or time intervals, as the event-driven or time-driven sensors make it ideal in critical energy-saving situations. The hybrid-driven sensors can be a mix of any of the aforementioned patterns, especially where a switch from time-driven to event-driven, or vice versa, has to occur when a particular event is triggered [9]. In WSNs, a node, typically a small, self-contained electronic device, is a crucial component equipped with sensors, a processor, a transceiver, and a power source. Nodes are engineered to be distributed throughout the physical environment of interest to sense, process, and wirelessly communicate data. Unfortunately, Node failure is common in WSNs, due to hardware failure, including the hardware components for both sensing, processing, and power source, or a software-related failure [10], [11]. Node failures caused by the sensor unit generate erroneous data, potentially leading to incorrect judgments by the network [12], [13]. These faulty sensor nodes can affect the service quality of WSNs, leading to false alarms (false positives) and missed detections (false negatives) [14]. Therefore, WSNs' reliability necessitates supplementation with algorithms to distinguish fault-affected events from the true ones [7], [15].

The fault-tolerance problem can be addressed through centralised, decentralised, or hybrid approaches. In the centralised approach, sensors report to the base station (BS) to identify faulty sensor nodes, which is considered the most straightforward approach but is limited in scalability. While, inthe decentralised approach, which requires less computation due to sensor resource constraints [16], [17], each sensor node should have an algorithm to identify the faulty nodes, and decision-making logics based on the neighbouring collaboration to determine whether to report to the BS or not and decide which sensor node to forward the report [18]. Non-classical logics, including fuzzy logics, Bayesian reasoning, probabilistic reasoning, paraconsistent logic, and machine learning, are particularly effective, as the presence of faulty sensors with the randomness introduced during deployment makes the traditional logics less suitable for handling uncertainties and inconsistencies. Fuzzy logic is used to select the next hub for packet routing decision by applying a series of fuzzy rules based on multiple criteria, such as energy, process capability, and distance, to assess whether a sensor node is not a faulty one and is suitable for sharing within the routing path [19], [20]. The Bayesian reasoning framework is applied to analyse the correlation between spatial and temporal data, identifying faulty sensor nodes that fail to trigger events within the specified time-based analysis range [21], [22]. The majority of voting algorithms utilise probabilistic reasoning and Monte Carlo simulations to minimise the impact of faulty sensor reports by fusing data from neighbouring sensor nodes [7], [8]. Paraconsistent logic permits inference from inconsistent information by handling the contradictions without leading to triviality (detailed in the next section). Machine learning makes approximate reasoning for fault node detection in WSNs through training models on sensor characteristics and reports to identify the patterns of actual events and detect the faulty node events [23], [24].

2. Fault-Tolerance and fault-detection approaches

Fault-tolerance in WSNs has been addressed through diverse approaches, including fuzzy logic, probabilistic reasoning, Bayesian inference, paraconsistent logic, and machine learning. While their implementation details vary, these methods share common principles such as evaluating node trustworthiness, aggregating neighbor data, and filtering faulty readings before event reports reach the base station. Table 1 summarises the main techniques, their operating principles, advantages, and implementation strategies.

Fuzzy-based methods [19], [20] adapt decision-making rules using parameters like residual energy, historical error rates, and processing time. For example, Nasurulla and R. Kaniezhil [19] proposed a Fuzzy-based Subordinate Support (FSS) system, where leader nodes (LNs) manage data aggregation and are periodically assessed for fitness; unfit LNs are replaced via subordinate nodes. Similarly, Talmale and Bhat [20] applied fuzzy rules to historical sensor metrics such as latency, packet loss, and error rates to identify and isolate defective nodes, enabling timely maintenance and improved reliability. Bayesian reasoning approaches [21], [22] leverage spatial-temporal correlations to dynamically assess trustworthiness. Wang and Liu [21] used a Bayesian Trust Model (BTM) with particle filtering to update trust indices and classify nodes as trusted/untrusted and event/non-event. Zhu and Sangaiah [22] extended this with weighted observations from "neighbourhood's neighbourhood" nodes, prioritising event nodes to improve boundary detection accuracy and mitigate the impact of errors.

Majority voting and neighbor cooperation schemes [8], [5], [7] address faults through consensus among neighboring nodes. Samanta et al. [8] introduced the True Event-Driven and Fault-Tolerant Routing (TED-FTR) algorithm, which confirms events through neighbor agreement and updates neighbor tables to exclude faulty nodes, achieving gains in latency, network lifetime, and packet error rate. Bhat and Santhosh [5] developed the Friendship Degree and Tenth Man Strategy (FD-TMS), which uses Monte Carlo simulations to validate event locations and refine fault identification, thereby enhancing detection reliability.

Machine learning methods [23], [24] classify node states from labeled datasets, offering adaptability to complex fault patterns. Prasad et al. [23] applied the XGBoost algorithm with regularisation and decision tree optimization to detect multiple fault types (spike, fixed bias, gain, out-of-bounds) using real-world sensor datasets. This approach achieved scalability, robustness, and high classification accuracy in energy-constrained networks.

By grouping related techniques, this review avoids redundancy and highlights how various strategies—whether rule-based, probabilistic, consensus-driven, or data-driven—share a common goal: improving fault tolerance and event detection reliability in WSNs.

Table 1. Summary listing of the developed Fault-Tolerance and fault-Detection Approaches

Reference	**Algorithm**	**Work Ground**	**Pros**	**Implementation**
[19]	Fuzzy-based Subordinate Support (FSS) system	Operates in chain-based and cluster-based WSNs structures with leader nodes managing data aggregation.	Ensures fault-tolerance, energy efficiency, and reliable data transmission in dynamic network environments.	It employs fuzzy logic processes (fuzzification, inference, defuzzification) and simulates them using the Mannasim framework.
[20]	Pre-fault detection mechanism based on a fuzzy rule-based method	Operates on collected metrics like temperature, error rates, and latency to evaluate node conditions.	Combining machine learning and fuzzy logic ensures efficient, accurate, and adaptable fault detection in dynamic environments.	The system integrates fuzzy rules for reasoning and machine learning for prediction to enhance decision-making.
[21]	Bayesian trust model (BTM)	Operates by assigning a trust index to each sensor node, evaluating its reliability based on spatiotemporal correlations within its community.	The algorithm efficiently updates trust values online using particle filtering, ensuring scalability and low computational overhead for large networks.	It integrates trust-based decision-making with distributed computation across nodes to detect events, even under sensor failures, and transmits results to sink nodes for region reconstruction.

Continued on next page

Table 1 – continued from previous page

Reference	Algorithm	Work Ground	Pros	Implementation
[22]	Distributed weighting fault-tolerance (DWFT)	The algorithm operates on the principle of spatial correlation among sensor nodes, emphasizing data exchange within a node's immediate and extended neighbourhood.	It improves fault detection accuracy and boundary node correction while reducing the likelihood of misclassification and enhancing overall event region detection.	The algorithm prioritises event nodes and uses weighting mechanisms and Bayesian estimation, ensuring efficient fault correction and robust detection performance.
[7]	True event-driven and fault-tolerant routing (TED-FTR)	The algorithm operates in event-driven WSNs, where nodes detect environmental events like wildfires or earthquakes while distinguishing true events from sensor faults through majority voting.	It ensures efficient energy usage, bypasses void areas, minimizes latency, and improves network lifetime by prioritising reliable data transmission paths.	The method integrates routing and event detection phases, leveraging neighbour state information, periodic beaconing, and distributed decision-making to achieve fault-tolerance and accurate event reporting.
[5]	Friendship degree and tenti mean strategy (FD-TMS)	Differentiates between true events and faulty readings in WSNs using distributed voting and topology validation.	Improves detection accuracy, minimizes false alarms, and reduces energy consumption.	Excludes faulty nodes based on trustworthiness and validates event location using the Monte Carlo Method.
[23]	XGBoost	XGBoost operates on gradient boosting principles, optimizing the loss function through second-order Taylor expansion and regularisation to avoid overfitting.	The algorithm is computationally efficient, achieving faster convergence, reduced iteration counts, and scalability for large-scale WSNs.	Train and test the XGBoost model, leveraging real-time datasets for fault detection and classification in WSN sensor nodes.

Continued on next page

Table 1 – continued from previous page

Reference	Algorithm	Work Ground	Pros	Implementation
[24]	Support Vector Machine, Artificial Neural Network (SVM-ANN)	The algorithms classify sensor data as faulty or fault-free based on labelled datasets collected from indoor, outdoor, single-hop, and multi-hop sensor scenarios.	Both machine learning methods achieve high classification accuracy (above 99%), with SVM slightly outperforming ANN in most scenarios.	The study compares the performance of the two algorithms to determine their suitability for enhancing sensor fault detection mechanisms in WSNs.

Continued on next page

This work proposes using paraconsistent logic (PL) in detecting the event decision-making process in the neighbours' cooperation method to prune the node from reporting that local shared neighbours between the decider and voter nodes do not support it. Application of machine learning to detect sensor faults in WSNs is examined on resource-constrained devices deployed in a region to collect environmental information, since inaccurate sensors may cause energy waste and loss of network bandwidth or power, thus requiring efficient fault-detection schemes. The approach regards fault detection as a binary classification, and through Kernel Support Vector Machine (KSVM) and Artificial Neural Network (ANN), it compares the classification performances of sensors as either faulty or okay. The proposed methods are investigated utilising annotated datasets containing indoor and outdoor environments with single-hop and multi-hop characteristics.

3. Method Development

3.1. Problem Definition

The network consists of N sensor nodes randomly deployed in a specific area, as shown in Figure 1. These sensors sense environmental data (e.g., temperature) and report to the BS when the sensor value exceeds a predetermined threshold (θ). When faulty nodes exist in a WSN, it can result in incorrect reports that reach the BS, negatively impacting the reliability and efficiency of the WSN. This problem is known as fault-tolerance in WSN, which requires different strategies to handle sending false reports to the BS and send only the true events that match real-world environment occurrences [7], [8]. The work of [7], [8] only considered the existence of a percentage of false positive sensor nodes in WSN without considering nodes generating false negative readings, which may impact the proposed algorithm for detecting true event strategy. This work considers false positive and false negative sensor nodes in the WSNs equally to avoid bias toward false positive sensors.

3.2. Proposed Method

3.2.1. Paraconsistent Logic (PL) A Main principle in Aristotelian logic is that two statements cannot be derived to be true or false [25], [26]. This assumption is known as the principle of contradiction. PL works when a sentence can be derived from the same premises to be true and false using different derivations [27], [28]. In PL, beyond the Boolean logic, other states are possible, such as inconsistent ($\top$) and incomplete ($\bot$) [29]. Paraconsistent Annotated Logic (PAL) is considered a practical variation of PL, which is based on a Hasse lattice to represent pieces of evidence of a statement P.

In this study, the term faulty nodes refers to any nodes that produce incorrect readings due to hardware, software, or communication failures. Faulty nodes are classified into two categories: (i) nodes generating false positive

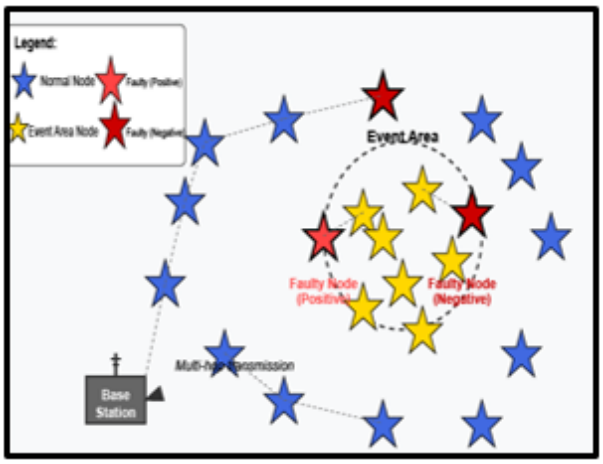

Figure 1. Wireless Sensor Network (WSN) model used in the study, showing node deployment, event detection zones, and communication ranges relevant to the proposed MV-PAL2V algorithm.

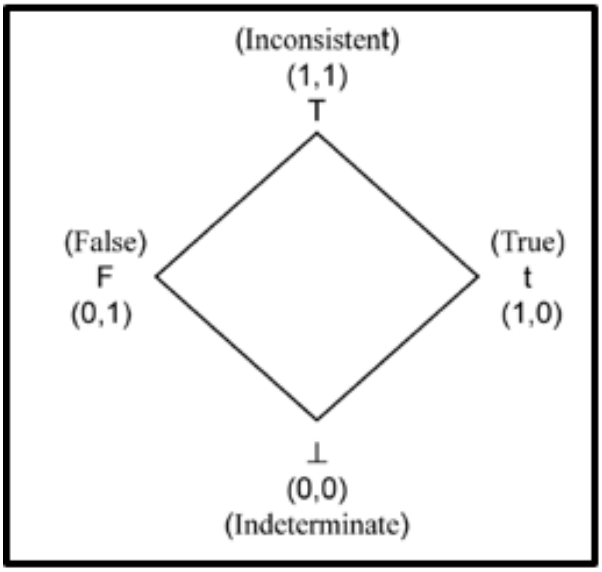

Figure 2. Four-vertex lattice representation of the Paraconsistent Annotated Logic with Two Values (PAL2V), mapping favorable (μ) and unfavorable (λ) evidence to the four extreme logical states [27].

readings, which incorrectly indicate the presence of an event, and (ii) nodes generating false negative readings, which fail to report an actual event. This terminology is used consistently throughout the manuscript to maintain clarity.

Paraconsistent Annotated Logic with Annotation of Two Values (PAL2V) uses two symbolic values μ and λ to express the degree of support for any statement P. PAL2V can be represented as an evidential atomic formula $P(\mu, \lambda)$, which makes up the annotation using a Hasse diagram (τ), consisting of a four-vertex lattice, as shown in the lattice diagram for PAL $\mathbb{E}_\tau$, Figure 2. The parameter (μ) that supports P is the favourable evidence degree, while (λ) that contradicts P is called the unfavourable evidence degree. Both μ and λ values are independent. For any $P(\mu, \lambda)$, the following intuitive cases may be concluded [29]:

- $P_T = (1, 0)$: indicates full support for evidence with no support for contradictory, i.e., truth of P;
- $P_F = (0, 1)$: indicates no support for evidence with full support for contradictory, i.e., falsehood of P;
- $P_\top = (1, 1)$: indicates full support for evidence with full support for contradictory, i.e., inconsistency of P;
- $P_\bot = (0, 0)$: indicates no support for evidence with no support for contradictory, i.e., incompleteness of P.

The direct application of the PAL2V gives what is called the Para-analyser or 12-state lattice concept, see Figure 3. Silva et al., 2011 [30] proposed dividing the PAL2V lattice into 12 states or regions, which results from splitting the four-vertex lattice into regions or logical states to support an expert system that can make decisions with a list of states and rules. These twelve states comprise the four main regions called extreme logical states symbolised as True (t), False (F), Inconsistent ($\top$), Paracomplete ($\bot$), and Q denotes the quasi state, and an additional eight states that may be summarised as follows [30, 31] and shown in Figure 3:

1. The Near True tending to Inconsistent (Qt-$\top$);
2. The Near True tending to Paracomplete (Qt-$\bot$);
3. The Near False tending to Inconsistent (QF-$\top$);
4. The Near False tending to Paracomplete (QF-$\bot$);
5. The Near Inconsistent tending to True (QT-t);
6. The Near Inconsistent tending to False (QT-F);
7. The Nearly Null Tending to True (Q$\bot$-t); and

8. The Nearly Null Tending to False (Q⊥-F).

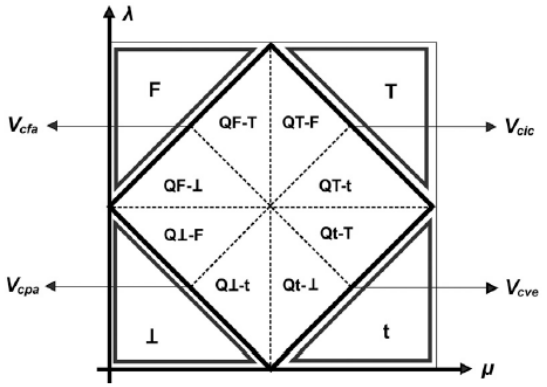

Figure 3. Para-Analyser 12-state lattice subdivision, illustrating how the basic PAL2V lattice is expanded into twelve logical states for finer decision-making in event detection [31].

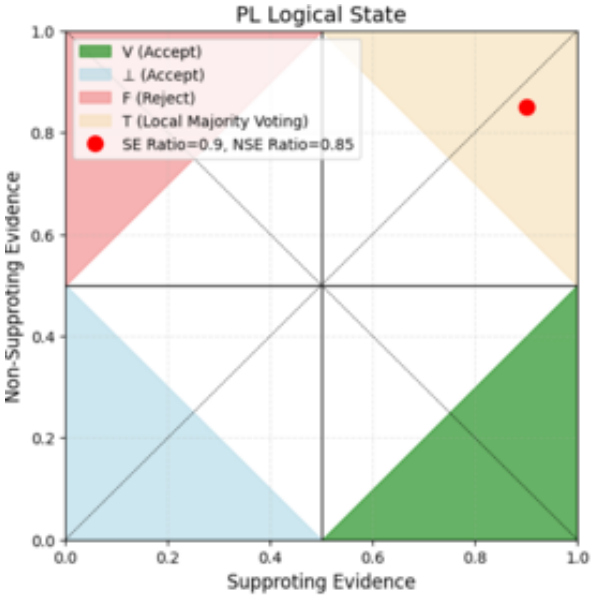

Figure 4. State lattice showing how supporting and contradicting evidence ratios are translated into PAL2V logical states, which guide the MV-PAL2V decision process.

To enhance the clarity of the 12-state lattice derived from PAL2V, we provide a simplified summary of each logical state along with the corresponding evidence conditions and their interpretations. This 12-state model extends the basic four logical states—True, False, Inconsistent, and Incomplete—into a finer granularity by incorporating quasi (Q) regions, which offer nuanced insights into borderline or transitional cases. The classification relies on two primary parameters: the degree of certainty (μ - λ) and the degree of contradiction ($\mu + \lambda$ - 1), both evaluated by the Para-Analyser. Table 2 presents a concise overview of these 12 logical states, specifying their symbolic representation, evidence conditions, and semantic meaning in the context of sensor decision-making. This clearer presentation facilitates understanding of how variations in supporting and contradicting evidence affect the classification and final actions taken in the MV-PAL2V algorithm.

Worked Example of μ and λ Mapping

Consider a deciding node K_i with four neighbors K_1, K_2, K_3, K_4, and K_5. Suppose K_2 and K_3 agree with K_i's reading (support), while K_4 and K_5 disagree (non-support). The favorable evidence is calculated as: $\mu = \frac{2}{4/2} = 1.0$ and the unfavorable evidence is: $\lambda = \frac{2}{4/2} = 1.0$ The degree of certainty is: The degree of certainty is $\mu - \lambda = 0.0$ and the degree of contradiction is $\mu + \lambda - 1 = 1.0$. This corresponds to the Inconsistent state in the 12-state lattice. This mapping allows each neighborhood voting outcome to be expressed as a logical state, guiding the MV-PAL2V decision process. The thresholds $c_1 = 0.5$ and $c_2 = -0.5$ were selected based on prior PAL2V literature and extensive pilot simulations, which demonstrated consistent separation of strong support, strong contradiction, and uncertain states across various network sizes and sensing modalities. While generally applicable, these thresholds can be tuned to fit application-specific data distributions.

The decision process of the Para-Analyser maps the calculated favorable (μ) and unfavorable (λ) evidence values to one of the twelve PAL2V logical states. This step ensures that each neighborhood voting outcome is expressed as a logical state, directly guiding the MV-PAL2V event decision process.

Algorithm 5 presents the Para-Analyser logical state evaluation. The algorithm takes μ and λ as inputs, computes the degrees of certainty and contradiction, and then assigns one of the twelve PAL2V logical states to guide MV-PAL2V decision-making.

Table 2. A concise overview of the 12 logical states.

State Description	**Symbol**	**Evidence Condition**	**Interpretation**
Completely True	t	$\mu = 1,\ \lambda = 0$ or $(\mu - \lambda) \geq 0.5$	Full support with no contradiction
Completely False	F	$\mu = 0,\ \lambda = 1$ or $(\mu - \lambda) \leq -0.5$	Full contradiction, no support
Inconsistent	$\top$	$(\mu + \lambda - 1) \geq 0.5$ and $-0.5 < (\mu - \lambda) < 0.5$	High conflict between supporting and contradicting evidence
Paracomplete (Incomplete)	$\bot$	$(\mu + \lambda - 1) \leq -0.5$ and $-0.5 < (\mu - \lambda) < 0.5$	Lack of sufficient information to draw a conclusion
Quasi-True tending to Inconsistent	Qt-$\top$	$(\mu - \lambda)$ slightly > 0, $(\mu + \lambda - 1) \approx 0.5$	Mostly true, but some strong contradictory evidence exists
Quasi-True tending to Paracomplete	Qt-$\bot$	$(\mu - \lambda)$ slightly > 0, $(\mu + \lambda - 1) \approx -0.5$	Mostly true, but not enough total evidence
Quasi-False tending to Inconsistent	QF-$\top$	$(\mu - \lambda)$ slightly < 0, $(\mu + \lambda - 1) \approx 0.5$	Mostly false, but contradictory evidence exists
Quasi-False tending to Paracomplete	QF-$\bot$	$(\mu - \lambda)$ slightly < 0, $(\mu + \lambda - 1) \approx -0.5$	Mostly false, but not enough total evidence
Quasi-Inconsistent tending to True	QT-t	$\mu > \lambda$, but $(\mu + \lambda - 1) \approx 0.5$	Evidence is closer to inconsistency but leans toward truth
Quasi-Inconsistent tending to False	QT-F	$\mu < \lambda$, but $(\mu + \lambda - 1) \approx 0.5$	Evidence is closer to inconsistency but leans toward falsehood
Quasi-Paracomplete tending to True	Q$\bot$-t	$\mu > \lambda$, but $(\mu + \lambda - 1) \approx -0.5$	Incomplete information, but evidence favors truth
Quasi-Paracomplete tending to False	Q$\bot$-F	$\mu < \lambda$, but $(\mu + \lambda - 1) \approx -0.5$	Incomplete information, but evidence favors falsehood

Although the integration of PAL2V into the majority voting process significantly enhances robustness against inconsistent or incomplete evidence, it introduces additional computational steps compared to traditional majority voting. These include calculating favorable (μ) and unfavorable (λ) evidence for each neighbor pair, evaluating degrees of certainty and contradiction, and mapping results to the 12-state lattice via the Para-Analyser. In dense network deployments, the per-node processing cost may increase proportionally with the number of neighbors, which could affect scalability in large-scale WSNs. However, the computational requirements remain lightweight relative to more complex machine learning approaches and are feasible for typical sensor node capabilities. For highly dense or resource-constrained networks, optimization strategies—such as precomputed lookup tables for PL state mapping, caching of frequently observed neighbor patterns, or limiting evaluation to a subset of high-trust neighbors—could further reduce processing overhead while maintaining decision quality.

3.2.2. Processing and Proposed Algorithms In this section, all algorithms used for initialising the WSN to send reports to the BS are listed with the proper explanations, and more attention is given to the majority voting – PAL2V (MV-PAL2V) algorithm, which is considered the work's main contribution. The system begins by sending

a Hello message from the BS. This message contains all the necessary information required for the various algorithms running on the sensor nodes. Since this work focuses on event detection, only the relevant event detection parameters are included in the Hello message. When a sensor node receives a Hello message, it sends its message to all its neighbours. This message contains the node's ID and the current number of known neighbours. Each time a sensor node receives a Hello message from another node, it increases its count of listed neighbours by one. The initialisation process is summarised in Algorithm 1. When a sensor node senses an event-related activity, it first clears the Reply column in the NB-Info table and checks whether the detected value exceeds the threshold θ. If the value is greater than θ, it sets the event detection flag to 1; otherwise, it sets it to 0. Then, it sends a Request message to all nodes listed in the NB-Info table. This event detection process is summarised in Algorithm 2. When a sensor receives a Request message after sensing the event, it creates a Reply message containing the sensor's ED flag, as illustrated in Algorithm 3. When a sensor receives a reply message from any neighbour sensor, it updates the corresponding sensor Reply flag in the NB-Info Table, Algorithm 4. The Para-Analyser (Algorithm 5) is considered the core algorithm in this work. It measures the degree of consistency between supporting and non-supporting evidence based on each sensor's Reply message. The algorithm returns the corresponding PL state using the supporting evidence μ and the non-supporting evidence λ. The possible PL logical states that can be reported are the extreme logical states: True (t), False (F), Inconsistent ($\top$), and Paracomplete ($\bot$), in addition to any Quasi state (Q). The MV-PAL2V algorithm, Algorithm 6, is the final step in the event detection process. After each node collects all Reply reports from its neighbours, it evaluates the degree of consistency between its own decision and the decisions shared by neighbouring nodes. Based on the Para-Analyser algorithm, the final decision is made as follows:

- State T (True): Accept the Reply report when the percentage of supporting nodes is high (e.g., 5 out of 6 neighbouring nodes support the sensor's decision).
- State F (False): Reject the Reply report when the percentage of support is low (e.g., only 1 out of 6 neighbours supports the sensor's decision).
- State I (Inconsistent): Apply majority voting among the shared neighbours when their responses conflict (e.g., 3 neighbours support the decision, and 2 do not).
- State $\bot$ or Q (Incomplete/Unknown): Mark the Reply report as incomplete or inconclusive when the number of shared replies is too low (e.g., only 2 neighbours responded), making it unreliable for a final decision.

Figure 4 shows how the PAL2V is applied to take the corresponding decision for the MV-PAL2V algorithm, where the supporting evidence is the ratio of shared neighbours between the deciding node and voting node that match the voting node's reply. The non-supporting evidence is the ratio of shared neighbours between the deciding node and voting node that do not match the voting node's reply. It shows that when the PL logic state is True, it supports the voting node decision. PyCharm is used to simulate the experiment and compare the results of the proposed MV-PAL2V with those of the standard majority voting proposed by [7], with the simulation parameters listed in Table '4. To improve clarity and enhance readability, the MV-PAL2V algorithm has been restructured into logically distinct components that reflect the step-by-step decision-making process. These components include: (1) event decision initialization, (2) evidence evaluation through support and non-support analysis, (3) para-analyser processing to determine the logical state, and (4) final event forwarding based on the output state. This structure provides better modularity and facilitates future extensions or adaptations. Additionally, the pseudocode has been reformatted with clearer indentation and consistent line structure to improve interpretation and implementation. By breaking the logic into manageable sections, the underlying decision rules of the proposed method are now easier to follow and verify.

The Para-Analyser plays a pivotal role in translating the collected neighbour feedback into a logical decision state using the principles of PAL2V. It evaluates the degree of certainty (μ - λ) and the degree of contradiction (μ + λ - 1) for each node based on the agreement or disagreement among shared neighbours' replies. Depending on these values, the Para-Analyser returns one of the twelve defined PAL2V logical states (e.g., True, False, Inconsistent, Incomplete, or a Quasi state). These outputs directly inform the final event decision in the MV-PAL2V algorithm:

- If the state is True, the node's event report is strongly supported and accepted.
- If False, the report is rejected due to lack of support.

- If Inconsistent, a local majority voting is applied to resolve contradictions.
- If Incomplete or a Quasi state, the decision is deferred or marked uncertain due to insufficient or ambiguous support.

This mechanism ensures that decisions are made with explicit consideration of both agreement and uncertainty, rather than relying solely on raw vote counts. This integration of logic reasoning improves robustness in noisy or partially faulty networks. The influence of each state is also embedded in the final decision scoring (FD_i) in Algorithm 6, allowing nuanced decision routing. In the proposed MV-PAL2V implementation, shared neighbors are identified based on overlapping communication ranges rather than random selection, ensuring that only physically reachable nodes participate in evidence aggregation. Incomplete or missing replies are treated as inconclusive inputs, which reduces the certainty value (μ - λ) but does not introduce false support or contradiction. While the simulation does not currently benchmark execution time or energy consumption, we note that PAL2V state mapping introduces minimal overhead relative to the message exchange process. A detailed runtime and energy consumption comparison with standard majority voting will be provided in future work. To support reproducibility, the simulation scripts and configuration files can be made available upon request. The main steps of both the baseline processing algorithms and the proposed MV-PAL2V approach are summarised in Table 3, providing a side-by-side comparison of their operational flow and key decision-making mechanisms.

4. Results and Discussion

4.1. Simulation Setup

In this work, PyCharm is used to simulate the experiment and compare the results of the proposed MV-PAL2V with those of the standard majority voting proposed by [7]. From 100 to 1000 nodes, with 100 nodes each step, are deployed randomly in (500, 500) meters. The transmission range is 60 meters, and the event range is 70 meters. The fault node percentage ranges from 5% to 20%, distributed equally between false alarms (false positives) and missed detections (false negatives) [15]. Normal sensor reading is represented by a normal distribution $\mathcal{N}(\mu_1, \sigma_1)$, for false positives $\mathcal{N}(\mu_2, \sigma_2)$, and false negatives $\mathcal{N}(\mu_3, \sigma_3)$. The experiment runs for 100 rounds; each round, the node and event locations are distributed randomly, and the average is reported. The simulation settings and parameters during the experiment are listed in Table 4.

The current evaluation assumes a uniform random distribution of faulty nodes, which simplifies baseline comparison but may not capture all real-world deployment conditions. In future experiments, spatially clustered fault models and dynamic event locations will be simulated to examine MV-PAL2V's resilience to heterogeneous conditions. Preliminary observations suggest that in sparse networks, reduced neighbor availability limits PAL2V's decision certainty, while in dense networks, abundant shared neighbors stabilize the decision process and improve FAR. The computational complexity of MV-PAL2V remains close to O(n) per decision, comparable to majority voting, but includes additional logic state evaluations; optimization strategies can further reduce this overhead.

4.2. Performance Metrics

The following metrics are related to event detection, which is the primary focus of this work:

a. Event node detection accuracy (ENDA): ENDA is the ratio between the number of nodes in the event area and the number of nodes that confirm the event within the event area [7], with higher ENDA values being desirable, reaching perfection at a value of 1.0 [32], [33], [34], [35], [36], [37], [38], [39], [40].

$$\text{ENDA} = \frac{\text{No. of event area nodes confirm event}}{\text{Total no. of nodes in the event area}} \tag{1}$$

b. False alarm rate (FAR): FAR is the ratio between the number of faulty nodes in the network and those reported to the BS, with lower FAR values being desirable, reaching perfection at a value of 0.0 [41], [42], [43], [44], [45], [46], [47], [48], [49].

$$\text{FAR} = \frac{\text{No. of faulty node reports to BS}}{\text{Total no. of positive faulty nodes in the network}} \tag{2}$$

Table 3. Processing and Proposed Algorithms

Algorithm 1 Node Initialisation Procedure for Node K_i

```
function INIT()
    if K_i is a Base Station (BS) then
        Transmit("HELLO" message)
    end if
end function
function ONHELLORECEIVED()
    if K_i detects a HELLO message from neighbour node K_j then
        Record Node ID in neighbour list
        Save the number of neighbours for Node ID in NB-Info Table
        BROADCASTHELLO
    end if
end function
function BROADCASTHELLO()
    for each K_j in NB-Info do
        Transmit("HELLO" message)
    end for
end function
```

Algorithm 2 Event Detection Process for Node K_i

```
function PERFORMSENSING()
    Clear Node K_i NB-Info Table Reply Column
    if DetectedValue ≥ Threshold θ then
        Set ED_i ← 1                          ▷ //Event detected
        for all K_i in NB-Info Table do
            Send "Request" message to K_i
        end for
    else
        Set ED_i ← −1                         ▷ //No event
    end if
end function
```

Algorithm 3 Process Request Message for Node K_j

```
function HANDLEREQUEST(INPUTMESSAGE)
    Prepare replyMessage:
        replyMessage.eventFlag ← E_{D_i}
        replyMessage.senderId ← K_i
        replyMessage.recipientId ← inputMessage.senderId
        replyMessage.type ← "REPLY"
    Send replyMessage to the requesting node
end function
```

Algorithm 4 Process Reply Message for Node K_j

```
function HANDLEREPLY(inputMessage)
    Find inputMessage.senderId record in NB-Info Table
    NB-Info.Reply ← inputMessage.eventFlag
end function
```

Algorithm 5 Para-Analyser Logical State Evaluation .

Input: μ (favorable evidence), λ (unfavorable evidence)

Algorithm 6 Modular Event Evaluation and Forwarding using MV-PAL2V

Continued on next page

Table 3 – continued from previous page

Output: PL_State (logical state from 12-state lattice)

// Calculate core PAL2V parameters
Certainty $\leftarrow \mu - \lambda$
Contradiction $\leftarrow \mu + \lambda - 1$
// Assign PL_State based on thresholds
if Certainty ≥ 0.5 **then**
  if Contradiction ≥ 0.5 **then**
    PL_State $\leftarrow$ "Qt-$\top$" ▷ Quasi-True tending to Inconsistent
  else if Contradiction ≤ -0.5 **then**
    PL_State $\leftarrow$ "Qt-$\bot$" ▷ Quasi-True tending to Paracomplete
  else
    PL_State $\leftarrow$ "t" ▷ Completely True
  end if
else if Certainty ≤ -0.5 **then**
  if Contradiction ≥ 0.5 **then**
    PL_State $\leftarrow$ "QF-$\top$" ▷ Quasi-False tending to Inconsistent
  else if Contradiction ≤ -0.5 **then**
    PL_State $\leftarrow$ "QF-$\bot$" ▷ Quasi-False tending to Paracomplete
  else
    PL_State $\leftarrow$ "F" ▷ Completely False
  end if
else if $-0.5 <$ Certainty < 0.5 **then**
  if Contradiction ≥ 0.5 **then**
    if $\mu > \lambda$ **then**
      PL_State $\leftarrow$ "QT-t" ▷ Quasi-Inconsistent tending to True
    else if $\mu < \lambda$ **then**
      PL_State $\leftarrow$ "QT-F" ▷ Quasi-Inconsistent tending to False
    else
      PL_State $\leftarrow$ "$\top$" ▷ Inconsistent
    end if
  else if Contradiction ≤ -0.5 **then**
    if $\mu > \lambda$ **then**
      PL_State $\leftarrow$ "QL-t" ▷ Quasi-Paracomplete tending to True
    else if $\mu < \lambda$ **then**
      PL_State $\leftarrow$ "QL-F" ▷ Quasi-Paracomplete tending to False
    else
      PL_State $\leftarrow$ "$\bot$" ▷ Paracomplete
    end if
  else
    PL_State $\leftarrow$ "Q" ▷ General Quasi state (uncertain)
  end if
end if
return PL_State

function EVALUATEEVENTDECISION()
  for each K_j in NB-Info Table **do**
    $FD_i \leftarrow 0$ ▷ Final Decision Score for Node K_i
    $Support_j, Non\_Support_j \leftarrow 0, 0$
    for each K_l in NB-Info Table **do**
      if K_l is a shared neighbour of K_i and K_j **then**
        if NB-Info.Reply$_l$ = NB-Info.Reply$_j$ **then**
          $Support_j \leftarrow Support_j + 1$ ▷ Agreement with K_j's report
        else
          $Non\_Support_j \leftarrow Non\_Support_j + 1$ ▷ Disagreement
        end if
      end if
    end for
    $\mu_j \leftarrow$ Support$_j$/(NumberOfNeighbours/2) ▷ Favorable evidence
    $\lambda_j \leftarrow \frac{\text{Non\_Support}_j}{(\text{NumberOfNeighbours}/2)}$ ▷ Unfavorable evidence
    $PL\_LS \leftarrow$ Para-Analyser(μ_j, λ_j) ▷ Get logical state
    ▷ Update the decision score based on logical state
    if $PL\_LS = F$ **then**
      $FD_i \mathrel{-}= ED_i$ ▷ Reject event decision due to contradiction
    else if $PL\_LS = t$ **then**
      $FD_i \mathrel{+}= ED_i$ ▷ Accept event decision with strong support
    else if $PL\_LS = T$ **then**
      if $\mu_j > \lambda_j$ **then**
        $FD_i \mathrel{+}= ED_i$ ▷ Inconsistent but leans toward support
      else
        $FD_i \mathrel{-}= ED_i$ ▷ Inconsistent but leans toward contradiction
      end if
    else if $PL\_LS = \bot$ or $PL\_LS = Q$ **then**
      $FD_i \mathrel{+}= ED_i$ ▷ Incomplete/uncertain but tentatively support
    end if
  end for
  ▷ Final forwarding decision based on net score
  if $FD_i > 0$ **then**
    Forward event report to BS through routing path
  end if
end function

Table 4. Simulation Parameters

Simulation Parameters	**Value**
Simulation area	500 m × 500 m
Total deployed nodes	100, 200, 300, 400, 500, 600, 700, 800, 900, 1000
Event detection radius	70 m
Node communication range	60 m
Events generated per round cycle	10 events
Standard sensor output (normal)	Gaussian (10,1)
Sensor output for true event	Gaussian (30,1)
False positive, faulty node sensor output	Gaussian (30,1)
False negative, faulty node sensor output	Gaussian (10,1)
Event detection threshold	25
Faulty node percentage	5%, 10%

4.3. Results Analysis

The following figures represent the analysis reports that use PL to enhance the majority voting algorithm by implementing PAL2V in the standard majority voting algorithm. Figure 5 represents the number of shared nodes between the deciding voting nodes. It shows that the number of shared nodes increases with a mild parabolic trend, demonstrating more opportunities for successfully applying the PAL2V approach to enhance majority voting. However, the figure explores that the relationship between the shared neighbours count and the number of nodes could be represented by a linear relationship (Shared Neighbours Count = 0.0512 x Number of nodes + 10.26, R^2 of 0.93). Moreover, two distinct plateaued regions could be observed with a steep transition at a number of nodes larger than five hundred.

Figure 6 presents a deeper analysis of PL logical states to determine the percentage of PL logical states used in voting decisions, revealing that approximately 12% of reports are not supported and are rejected, reducing ENDA and FAR errors. Figures 7 and 8 compare ENDA for benchmark work (TED-FTR) and the MV-APL2v algorithm, for different percentages of faulty nodes for false alarms (False positives) and missed detections (False negatives). The figures explore tiny ENDA values at a low number of nodes (i.e., 100) due to the lower density of nodes participating in the voting process. This trend points out the importance of the minimum number of nodes for the algorithm to be efficient, i.e., the performance metrics of the proposed algorithm get improved with the increase in the number of nodes, which implies reducing the impact of non-supporting voters. Moreover, ENDA exhibits higher values for MV-PAL2V than that of TED-FTR at 5% and 10% faulty nodes at all numbers of nodes, confirming the performance enhancement of the proposed algorithm compared to the benchmarked one.

Figures 9 and 10 compare FAR for benchmark work (TED-FTR) and the MV-PAL2V algorithm, for different percentages of faulty nodes for both false alarms (False positives) and missed detections (False negatives). The figures explore a somewhat different FAR trend compared to ENDA's. The FAR values show a flattened bell-shaped curve with an elongated right tail at a number of nodes larger than five hundred for 5% and 10% faulty nodes. At a low number of nodes, the low density of nodes participating in the voting process and sending to the BS is not enough for both algorithms to perform efficiently.

With the increase in the number of nodes, the FAR increases due to activation of the voting process. However, when the number of nodes continues to increase up to 600 nodes, the FAR decreases again due to the voting process toward a correct decision. The MV-PAL2V performed similarly to that of TED-FTR in terms of FAR.

The observed peaks and dips in performance metrics such as ENDA and FAR across varying node counts can be attributed to the changing network density and its effect on neighborhood formation. At lower node densities (e.g., 100–300 nodes), the number of neighbors per node is relatively limited, which reduces the reliability of majority

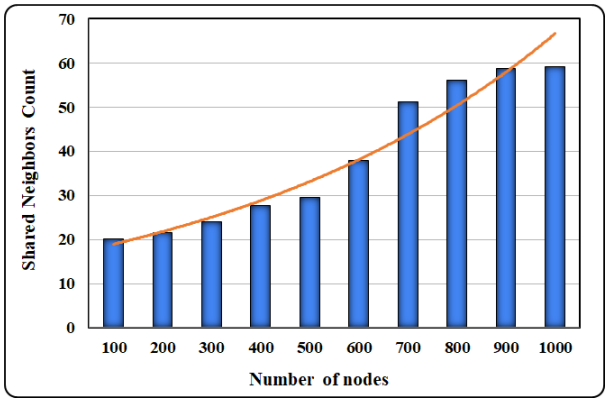

Figure 5. Relationship between the number of nodes in the network and the count of shared neighbors between deciding and voting nodes, indicating the potential for effective PAL2V application.

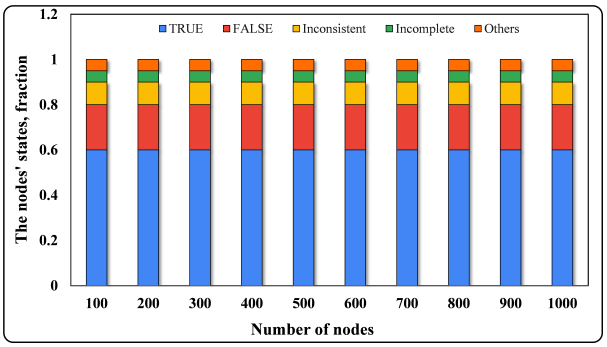

Figure 6. Distribution of PAL2V logical states observed during simulations, showing the proportion of reports accepted, rejected, or marked uncertain, and their effect on ENDA and FAR.

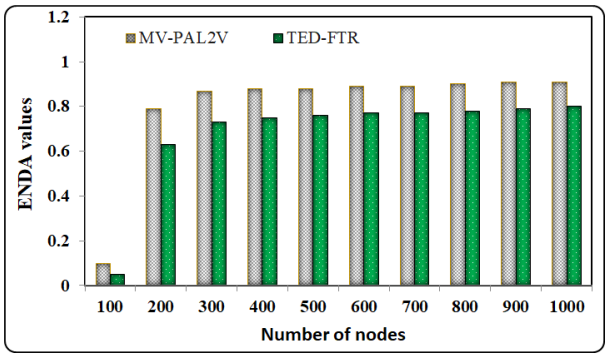

Figure 7. Comparison of Event Node Detection Accuracy (ENDA) between TED-FTR and MV-PAL2V algorithms for 5% faulty nodes, demonstrating the performance improvement of the proposed method.

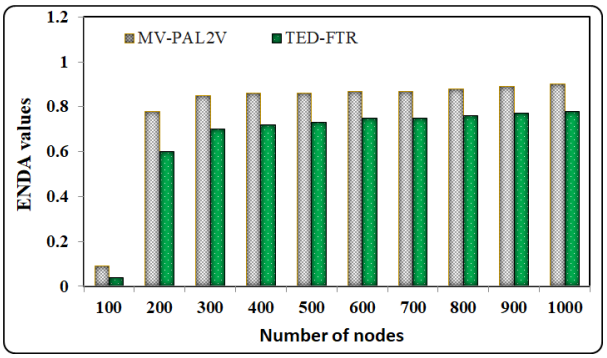

Figure 8. Comparison of ENDA between TED-FTR and MV-PAL2V algorithms for 10% faulty nodes, showing consistent gains across different network sizes.

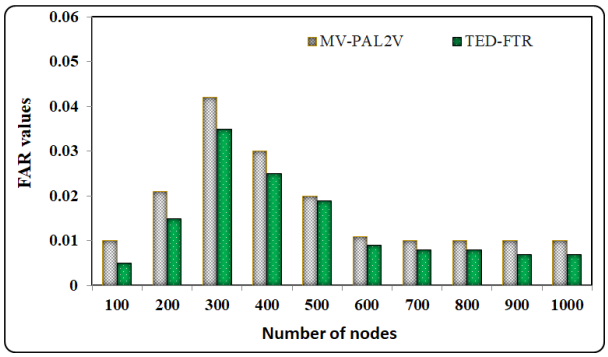

Figure 9. Comparison of False Alarm Rate (FAR) between TED-FTR and MV-PAL2V algorithms for 5% faulty nodes, highlighting how increased network density reduces FAR.

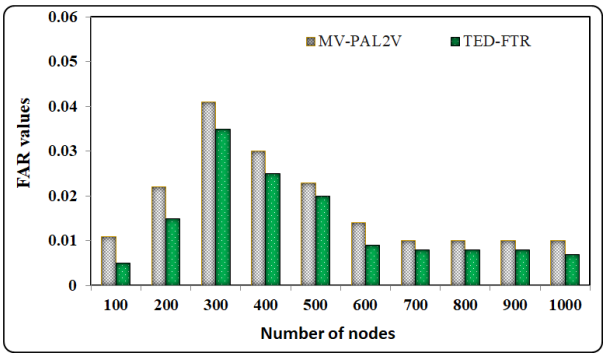

Figure 10. Comparison of FAR trends for MV-PAL2V and TED-FTR algorithms across different node densities and faulty node percentages.

voting and the application of PAL2V due to insufficient shared evidence. As the number of nodes increases (e.g.,

400–600), the network transitions into a denser configuration, improving neighbor connectivity and enabling more accurate support and contradiction analysis. However, this transitional range may also introduce variability due to uneven clustering or localized redundancy, resulting in temporary performance dips. Beyond 600 nodes, the network becomes sufficiently dense, stabilizing the neighbor relationships and yielding improved and consistent event detection accuracy and reduced false alarms. These nonlinear behaviors emphasize the sensitivity of logic-based voting schemes to the underlying network topology and highlight the importance of maintaining adequate node density in WSN deployments.

While the current evaluation uses multiple simulation rounds and includes correlation-based analysis to validate the performance of the proposed MV-PAL2V algorithm, formal statistical significance testing—such as t-tests, confidence intervals, or ANOVA—was not conducted in this version. This omission is primarily due to the simulation-focused nature of the study and the deterministic control over experimental parameters. However, we acknowledge the importance of statistically validating the observed improvements in ENDA and FAR. Incorporating hypothesis-driven statistical methods to compare MV-PAL2V against benchmark algorithms like TED-FTR will be prioritized in future work, especially in settings involving real-world datasets or hardware testbeds, where stochastic effects are more pronounced.

4.4. Step-by-Step Example of PAL2V Application

To illustrate how PAL2V enhances decision-making over traditional majority voting, we provide a step-by-step example using a simplified 5-node scenario. This illustrative case demonstrates how the proposed MV-PAL2V algorithm interprets the local neighborhood evidence, applies paraconsistent logic, and arrives at a final decision that is more robust to inconsistencies than a basic voting approach.

Consider a simplified wireless sensor network comprising five sensor nodes: K1 (deciding node), and its neighbours K2, K3, K4, and K5. Assume K1 detects an event and requests voting feedback from its neighbours. The neighbor responses used for the simplified 5-node example are summarised in Table 5, showing the voting replies received by node K1 from its immediate neighbours. Each neighbour responds with either:

- 1 = supports event detection,
- -1 = does not support event detection.

The replies received by K1 are as follows:

Table 5. Replies received by node K1 from neighbouring nodes (K2–K5) indicating support (+1) or rejection (-1) of the detected event.

Neighbour	**Reply**
K2	1
K3	1
K4	-1
K5	-1

This yields:

- Supporting replies (μ) = 2 (K2, K3)
- Non-supporting replies (λ) = 2 (K4, K5)

Assuming K1 has 4 shared neighbours, the evidence degrees are calculated as:

- $\mu = 2/(4/2) = 1.0$
- $\lambda = 2/(4/2) = 1.0$

The Para-Analyser computes:

- Degree of Certainty = $\mu - \lambda = 0$
- Degree of Contradiction = $\mu + \lambda - 1 = 1$

This falls into the Inconsistent ($\top$) state according to the 12-state lattice.

Now applying MV-PAL2V logic:

- Since $\mu = \lambda$, the inconsistency is unresolved.
- The algorithm defers to a local majority voting, which results in a tie (2 vs. 2).
- Depending on implementation policy (e.g., tie-breaking or confidence threshold), the node may choose to withhold reporting, **or** mark the event as uncertain.

In contrast, a traditional majority voting algorithm may:

- Simply reject the event (if it requires $>$50% agreement),
- Or fail to model the uncertainty explicitly, leading to possible misclassification.

This example highlights PAL2V's ability to:

- Model the conflict explicitly,
- Adapt decisions to the evidence context, and
- Avoid false alarms or missed detections that may arise under simple voting rules.

4.5. Compound Desirability Range

The compound desirability range describes the range that yields the lowest FAR and highest ENDA according to the number of nodes and the shared neighbours count. Figure 11 is the bubble plot representing these four variables, which is directly readable and interpretable regarding FAR and ENDA. It demonstrates the crucial role that the shared neighbours count play. The bubbles with the smaller sizes appear only at the higher values of the shared neighbours count, except for the lowest number of nodes, i.e., at 100. The matrix plot shown in Figure 12 demonstrates the Spearman correlation at a 95% confidence interval among the interplaying variables. Note that it shows only the interactions at 5% faulty nodes for clarity, as 5% and 10% faulty nodes behave much the same. It is emphasised that the effect of the number of nodes and the shared neighbours on the FAR values has a -0.762 negative coefficient, indicating an inverse coherence between both. The Spearman correlation coefficient for ENDA is 0.562, with the number of nodes and the shared neighbours, indicating a direct but less coherent relationship. The Spearman coefficient between the number of nodes and shared neighbour count is 1, emphasising direct and complete coherence. Moreover, Figures 13 and 14 show the counterplot of the FAR and ENDA against the number of nodes and the shared neighbours count, respectively. Figures 13 and 14 show areas of desirability ranges that yield the best performance metrics, i.e., lowest FAR and highest ENDA. Finally, Figure 15 shows much the same behaviour, indicating the area of lowest FAR lies at the top right corner, i.e., at the highest number of nodes and shared neighbours count. All these themes emphasise that the number of nodes plays a crucial role in the reliability of the WSNs' data, and hence, trade-offs and reliable fault tolerance processing algorithms are mandatory for the WSNs to be well posed.

5. Conclusion and Future Work

This study employs the PL approach to address the fault-tolerance event detection problem in WSNs, where majority voting, including neighbour cooperation, is adopted. Our proposed approach analyses the inconsistency between reports from voter nodes, based on their data and the local data shared with the deciding node. This approach proved particularly beneficial for prune node reports not supported by other local voters' reports shared by the deciding node, resulting in enhanced performance in terms of ENDA and FAR. The approach can be applied to different neighbours' cooperation methods by first propagating data that can be used in the decision-making process, allowing other nodes to implement the voter algorithm on the local data shared between the voter and

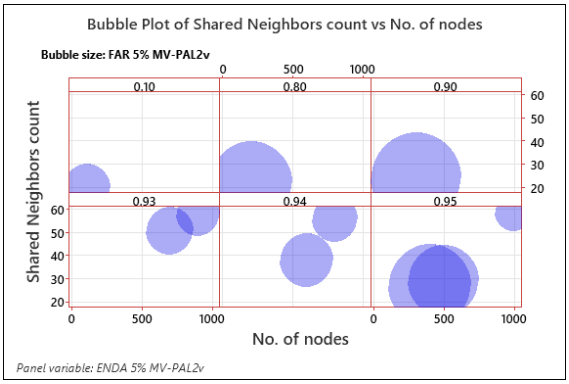

Figure 11. Bubble plot of compound desirability range, illustrating optimal regions for low FAR and high ENDA as functions of node count and shared neighbor count.

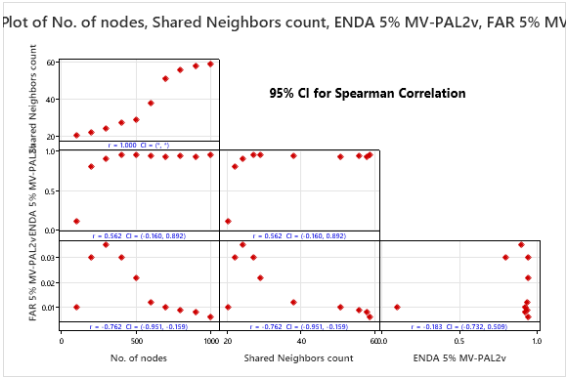

Figure 12. Matrix plot showing Spearman correlation coefficients among number of nodes, shared neighbor count, ENDA, and FAR, revealing the strength and direction of variable relationships.

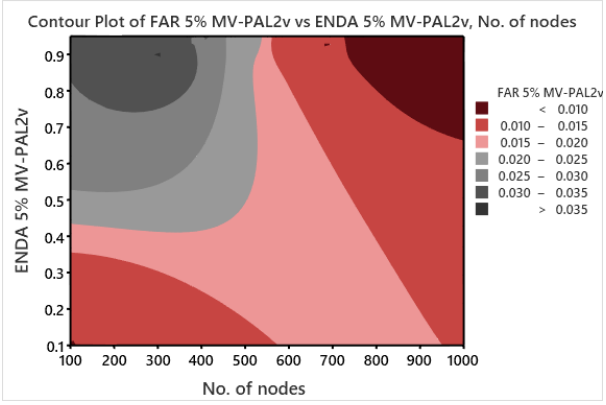

Figure 13. Contour plot of FAR versus number of nodes and ENDA, identifying regions where both metrics achieve desirable values.

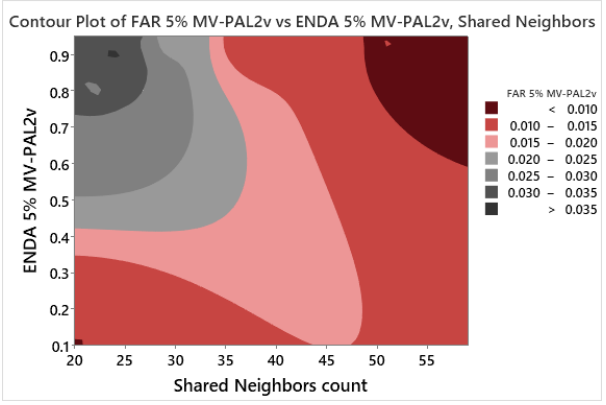

Figure 14. Contour plot of FAR versus shared neighbor count and ENDA, highlighting the influence of neighbor availability on detection reliability.

the decider, which promotes the voting process by pruning non-supporting reports. The statistical and interaction analysis emphasised that the number of nodes plays a crucial role in the reliability of the WSNs' data, and hence, trade-offs and reliable fault tolerance processing algorithms are mandatory for the WSNs to be well posed. In

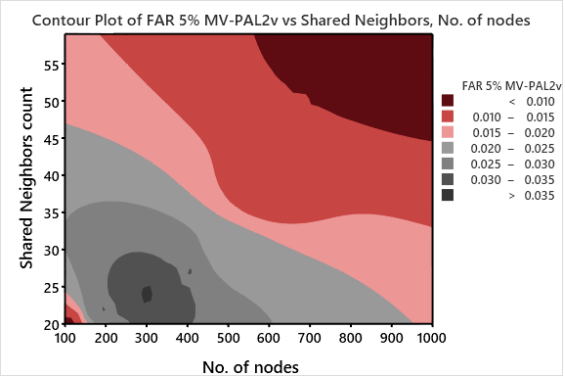

Figure 15. Contour plot of FAR versus number of nodes and shared neighbor count, emphasizing that optimal performance occurs at high node density and shared neighbor values.

applications involving sensitive data (e.g., healthcare), neighbor data sharing must be managed to prevent leakage of personal information. Privacy-preserving aggregation techniques, such as secure multiparty computation or anonymization, could be integrated with MV-PAL2V to ensure compliance with ethical and regulatory standards. While this study evaluates MV-PAL2V in a temperature sensing context, the approach is agnostic to the sensing modality and can be extended to vibration, humidity, or multi-modal sensor networks, provided that neighbor readings can be quantified as favorable or unfavorable evidence. Future extensions will explore robustness against adversarial conditions, such as Byzantine faults or malicious nodes intentionally skewing evidence to mislead the decision process.

Future extensions of this research will focus on adapting the MV-PAL2V algorithm to more complex and dynamic environments such as mobile wireless sensor networks (MWSNs) and edge-assisted IoT systems, where node mobility and real-time decision-making introduce new challenges. We also intend to implement the framework in physical testbeds to validate its performance under real-world operating conditions. Additionally, future efforts will incorporate statistical significance tests (e.g., t-tests, confidence intervals) to strengthen empirical validation. Given the potential computational burden of evaluating PL states at scale, we aim to explore optimization techniques, such as precomputed logical state tables or adaptive neighborhood filtering, to enhance the algorithm's efficiency and scalability in dense deployments. Hybrid approaches, such as integrating PAL2V with machine learning classifiers (e.g., XGBoost), may further reduce FAR while retaining interpretability, providing an avenue for combining symbolic logic and statistical learning.

REFERENCES

1. R. E. Mohamed, A. I. Saleh, M. Abdelrazzak, and A. S. Samra, "Survey on wireless sensor network applications and energy efficient routing protocols," *Wireless Personal Communications*, vol. 101, no. 2, pp. 1019–1055, 2018.
2. J. Zhu, M. Ullah, S. Ullah, M. B. Riaz, A. B. Saqib, A. M. Alamri, and S. A. AlQahtani, "A novel numerical solution of nonlinear stochastic model for the propagation of malicious codes in wireless sensor networks using a high order spectral collocation technique," *Scientific Reports*, vol. 15, no. 1, p. 228, 2025.
3. H. Hayat, T. Griffiths, D. Brennan, R. P. Lewis, M. Barclay, C. Weirman, B. Philip, and J. R. Searle, "The state-of-the-art of sensors and environmental monitoring technologies in buildings," *Sensors*, vol. 19, no. 17, p. 3648, 2019.
4. M. Ali, B. Salah, and T. Habib, "Utilizing industry 4.0-related technologies and modern techniques for manufacturing customized products–smart yogurt filling system," *Journal of Engineering Research*, vol. 12, no. 3, pp. 468–475, 2024.
5. S. J. Bhat and S. KV, "A localization and deployment model for wireless sensor networks using arithmetic optimization algorithm," *Peer-to-Peer Networking and Applications*, vol. 15, no. 3, pp. 1473–1485, 2022.
6. J. Yang, M. Xu, W. Zhao, and B. Xu, "A multipath routing protocol based on clustering and ant colony optimization for wireless sensor networks," *Sensors*, vol. 10, no. 5, pp. 4521–4540, 2010.
7. P. Biswas and T. Samanta, "True event-driven and fault-tolerant routing in wireless sensor network," *Wireless Personal Communications*, vol. 112, no. 1, pp. 439–461, 2020.
8. G. H. Adday, S. K. Subramaniam, Z. A. Zukarnain, and N. Samian, "Friendship degree and tenth man strategy: A new method for differentiating between erroneous readings and true events in wireless sensor networks," *IEEE Access*, vol. 11, pp. 127651–127668,

2023.
9. G. Sahar, K. A. Bakar, S. Rahim, N. A. K. K. Khani, and T. Bibi, "Recent advancement of data-driven models in wireless sensor networks: a survey," *Technologies*, vol. 9, no. 4, p. 76, 2021.
10. G. H. Adday, S. K. Subramaniam, Z. A. Zukarnain, and N. Samian, "Fault tolerance structures in wireless sensor networks (wsns): Survey, classification, and future directions," *Sensors*, vol. 22, no. 16, p. 6041, 2022.
11. R. R. Swain, P. M. Khilar, and S. K. Bhoi, "Underlying and persistence fault diagnosis in wireless sensor networks using majority neighbors co-ordination approach," *Wireless Personal Communications*, vol. 111, no. 2, pp. 763–798, 2020.
12. M. Yehia Habash, N. M. A. E. Ayad, and A. E. A. E. Ammar, "Fault tolerant radiation monitoring system using wireless sensor and actor network in a nuclear facility," *International Journal of Electronics and Telecommunications*, vol. 67, no. 1, pp. 87–93, 2021.
13. G. Jesus, A. Casimiro, and A. Oliveira, "Using machine learning for dependable outlier detection in environmental monitoring systems," *ACM Transactions on Cyber-Physical Systems*, vol. 5, no. 3, pp. 1–30, 2021.
14. M.-C. Wueng and S.-I. Hwang, "On surveillance quality of sensor networks with faulty sensor nodes," in *The 9th International Conference on Advanced Communication Technology*, vol. 1, pp. 634–639, IEEE, 2007.
15. M. Safaei, A. S. Ismail, H. Chizari, M. Driss, W. Boulila, S. Asadi, and M. Safaei, "Standalone noise and anomaly detection in wireless sensor networks: a novel time-series and adaptive bayesian-network-based approach," *Software: Practice and Experience*, vol. 50, no. 4, pp. 428–446, 2020.
16. D. G. Takale, P. N. Mahalle, and B. Sule, "Centralized and distributed approach: A comparative analysis of fault diagnosis approaches in wireless sensor networks," in *Machine Learning for Environmental Monitoring in Wireless Sensor Networks*, pp. 247–268, IGI Global, 2025.
17. K. L. Man, C. Chen, and D. Hughes, "Decentralized fault detection and management for wireless sensor networks," in *2010 5th International Conference on Future Information Technology*, pp. 1–6, IEEE, 2010.
18. D. G. Takale, P. N. Mahalle, and B. Sule, "Centralized and distributed approach: A comparative analysis of fault diagnosis approaches in wireless sensor networks," in *Machine Learning for Environmental Monitoring in Wireless Sensor Networks*, pp. 247–268, IGI Global, 2025.
19. I. Nasurulla and R. Kaniezhil, "Integration of fault-tolerant feature to omieepb routing protocol in wireless sensor network," *International Journal of Intelligent Computing and Cybernetics*, vol. 15, no. 3, pp. 414–424, 2022.
20. R. Talmale and M. N. Bhat, "Energy attentive and pre-fault recognize mechanism for distributed wireless sensor network using fuzzy logic approach," *Wireless Personal Communications*, vol. 124, no. 2, pp. 1263–1280, 2022.
21. J. Wang and B. Liu, "Online fault-tolerant dynamic event region detection in sensor networks via trust model," in *2017 IEEE wireless communications and networking conference (WCNC)*, pp. 1–6, IEEE, 2017.
22. Y. Xiang, H. Li, Z. Xie, and P. Li, "Distributed weighting fault-tolerant algorithm for event region detection in wireless sensor networks," in *2008 International Conference on Communications, Circuits and Systems*, pp. 414–418, IEEE, 2008.
23. R. Prasad, R. Baghel, and P. V. Sanjay, "Xgboost based fault detection for wireless sensor networks," in *2024 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, pp. 1–5, IEEE, 2024.
24. A. Abdulkarim, I. E. Ehile, and R. C. Kizilirmak, "Utilizing machine learning for sensor fault detection in wireless sensor networks," in *2024 26th International Conference on Advanced Communications Technology (ICACT)*, pp. 343–349, IEEE, 2024.
25. A. C. Varzi, "Logic, ontological neutrality," *Contradictions: Logic, History, Actuality*, vol. 6, p. 53, 2014.
26. X. Liu, P. Besnard, and S. Doutre, "Paraconsistent inference relations induced from inconsistency measures," *International Journal of Approximate Reasoning*, vol. 152, pp. 183–197, 2023.
27. N. Costa, J. M. Abe, and U. Subrahmanian, "Remarks on annotated logic," 1991.
28. S. Jaśkowski, "Propositional calculus for contradictory deductive systems (communicated at the meeting of march 19, 1948)," *Studia Logica: An International Journal for Symbolic Logic*, vol. 24, pp. 143–160, 1969.
29. J. I. Da Silva Filho, "Treatment of uncertainties with algorithms of the paraconsistent annotated logic," *Journal of Intelligent Learning Systems and Applications*, vol. 4, no. 2, pp. 144–153, 2012.
30. J. I. Da Silva Filho, G. Lambert-Torres, L. F. P. Ferrara, M. C. Mário, M. R. dos Santos, A. S. Onuki, J. de Melo Camargo, and A. Rocco, "Paraconsistent algorithm extractor of contradiction effects-paraextrctrctr," *Journal of Software Engineering and Applications*, vol. 4, no. 10, pp. 579–584, 2011.
31. H. M. Côrtes, P. E. Santos, and J. I. da Silva Filho, "Monitoring electrical systems data-network equipment by means of fuzzy and paraconsistent annotated logic," *Expert Systems with Applications*, vol. 187, p. 115865, 2022.
32. R. Farouk, M. Elsayed, and M. Aly, "Medical image denoising based on log-gabor wavelet dictionary and k-svd algorithm," *International Journal of Computer Applications*, vol. 141, no. 1, pp. 27–32, 2016.
33. E. Elsayed, D. Salem, and M. Aly, "A fast image denoising algorithm based on tls model and sparse representation," *International Journal of imaging and robotics*, vol. 19, no. 2, 2019.
34. E. Elsayed and M. Aly, "Hybrid between ontology and quantum particle swarm optimization for segmenting noisy plant disease image," *International Journal of Intelligent Engineering and Systems*, vol. 12, no. 5, pp. 299–311, 2019.
35. E. K. Elsayed, D. R. Salem, and M. Aly, "A fast quantum particle swarm optimization algorithm for image denoising problem.," *International Journal of Intelligent Engineering & Systems*, vol. 13, no. 1, 2020.
36. M. Aly and N. S. Alotaibi, "A novel deep learning model to detect covid-19 based on wavelet features extracted from mel-scale spectrogram of patients' cough and breathing sounds," *Informatics in Medicine Unlocked*, vol. 32, p. 101049, 2022.
37. M. Aly and N. S. Alotaibi, "A new model to detect covid-19 coughing and breathing sound symptoms classification from cqt and mel spectrogram image representation using deep learning," *Int. J. Adv. Comput. Sci. Appl*, vol. 13, pp. 601–611, 2022.
38. M. Aly and A. S. Alotaibi, "Molecular property prediction of modified gedunin using machine learning," *Molecules*, vol. 28, no. 3, p. 1125, 2023.
39. A. Hjazi, Y. Q. Almajidi, W. R. Kadhum, M. Aly, J. Malviya, M. N. Fenjan, A. Alawadi, A. Alsaalamy, A. Chandramauli, and L. Baharinikoo, "Optimization of removal of sulfonamide antibiotics by magnetic nanocomposite from water samples using central composite design," *Water Resources and Industry*, vol. 30, p. 100229, 2023.

40. M. Aly and A. S. Alotaibi, "Emu-net: Automatic brain tumor segmentation and classification using efficient modified u-net.," *Computers, Materials & Continua*, vol. 77, no. 1, 2023.
41. M. Aly, A. Ghallab, and I. S. Fathi, "Enhancing facial expression recognition system in online learning context using efficient deep learning model," *IEEE Access*, vol. 11, pp. 121419–121433, 2023.
42. M. H. Behiry and M. Aly, "Cyberattack detection in wireless sensor networks using a hybrid feature reduction technique with ai and machine learning methods," *Journal of Big Data*, vol. 11, no. 1, p. 16, 2024.
43. M. Aly, A. Ghallab, and I. S. Fathi, "Tumor vit-gru-xai: advanced brain tumor diagnosis framework: vision transformer and gru integration for improved mri analysis: a case study of egypt," *IEEE Access*, vol. 12, pp. 184726–184754, 2024.
44. M. Aly, "Revolutionizing online education: Advanced facial expression recognition for real-time student progress tracking via deep learning model," *Multimedia Tools and Applications*, pp. 1–40, 2024.
45. M. Aly, "Weakly-supervised thyroid ultrasound segmentation: Leveraging multi-scale consistency, contextual features, and bounding box supervision for accurate target delineation," *Computers in Biology and Medicine*, p. 109669, 2025.
46. M. Aly and A. S. Alotaibi, "Hybrid butterfly-grey wolf optimization (hb-gwo): A novel metaheuristic approach for feature selection in high-dimensional data," *Statistics, Optimization & Information Computing*, vol. 13, no. 6, pp. 2575–2600, 2025.
47. M. Aly and I. S. Fathi, "Recognizing american sign language gestures efficiently and accurately using a hybrid transformer model," *Scientific Reports*, vol. 15, no. 1, p. 20253, 2025.
48. I. S. Fathi, H. Ardah, G. Hassan, and M. Aly, "Protecting iot networks through ai-based solutions and fractional tchebichef moments," *Fractal and Fractional*, 2025.
49. M. Aly and M. H. Behiry, "Enhancing anomaly detection in iot-driven factories using logistic boosting, random forest, and svm: A comparative machine learning approach," *Scientific Reports*, vol. 15, no. 1, p. 23694, 2025.